Structural characterizations of the navigational expressiveness of relation algebras on a tree[☆]

George H. L. Fletcher[a,*], Marc Gyssens[b], Jan Paredaens[c], Dirk Van Gucht[d], Yuqing Wu[d]

[a] *Eindhoven University of Technology, P.O. Box 513, 5600 MB Eindhoven, The Netherlands*
[b] *Hasselt University and Transnational University of Limburg, School for Information Technology, Belgium*
[c] *University of Antwerp, Belgium*
[d] *Indiana University, Bloomington, USA*

Abstract

Given a document D in the form of an unordered node-labeled tree, we study the expressiveness on D of various basic fragments of XPath, the core navigational language on XML documents. Working from the perspective of these languages as fragments of Tarski's relation algebra, we give characterizations, in terms of the structure of D, for when a binary relation on its nodes is definable by an expression in these algebras. Since each pair of nodes in such a relation represents a unique path in D, our results therefore capture the sets of paths in D definable in each of the fragments. We refer to this perspective on language semantics as the "global view." In contrast with this global view, there is also a "local view" where one is interested in the nodes to which one can navigate starting from a particular node in the document. In this view, we characterize when a set of nodes in D can be defined as the result of applying an expression to a given node of D. All these definability results, both in the global and the local view, are obtained by using a robust two-step methodology, which consists of first characterizing when two nodes cannot be distinguished by an expression in the respective fragments of XPath, and then bootstrapping these characterizations to the desired results.

Keywords: trees, relation algebra, XML, XPath, bisimulation, instance expressivity

☆A preliminary version of some of the results given here were presented at the Twenty-Fifth ACM Symposium on Principles of Database Systems [1].

*Corresponding author. Phone: +31 (0)40 247 26 24. Fax: +31 (0)40 243 66 85.
Email addresses: `g.h.l.fletcher@tue.nl` (George H. L. Fletcher), `marc.gyssens@uhasselt.be` (Marc Gyssens), `jan.paredaens@ua.ac.be` (Jan Paredaens), `vgucht@cs.indiana.edu` (Dirk Van Gucht), `yuqwu@indiana.edu` (Yuqing Wu)

1. Introduction

In this paper, we investigate the expressive power of several basic fragments of Tarski's relation algebra [2] on finite tree-structured graphs. Tarski's algebra is a fundamental tool in the field of algebraic logic which finds various applications in computer science [3–6]. Our investigation is specifically motivated by the role the relation algebra plays in the study of database query languages [7–13]. In particular, the algebras we consider in this paper correspond to natural fragments of XPath. XPath is a simple language for navigation in XML documents (i.e., a standard syntax for representing node-labeled trees), which is at the heart of standard XML transformation languages and other XML technologies [14]. Keeping in the spirit of XML, we will continue to speak in what follows of trees as "documents" and the algebras we study as "XPath" algebras.

XPath can be viewed as a query language in which an expression associates to every document a binary relation on its nodes representing all navigation paths in the document defined by that expression [9, 15, 16]. From this query-level perspective, several natural semantic issues have been investigated in recent years for various fragments of XPath. These include expressibility, closure properties, and complexity of evaluation [8, 9, 15, 17, 18], as well as decision problems such as satisfiability, containment, and equivalence [19–21].

Alternatively, we can view XPath as a navigational tool on a particular given document, and study expressiveness issues from this document-level perspective. (A similar duality exists in the relational database model, where Bancilhon [22] and Paredaens [23] considered and characterized expressiveness at the instance level, which, subsequently, Chandra and Harel [24] contrasted with expressiveness at the query level.)

In this setting, our goal is to characterize, for various natural fragments of XPath, when a binary relation on the nodes of a given document (i.e., a set of navigation paths) is definable by an expression in the fragment.

To achieve this goal, we develop a robust two-step methodology. The first step consists of characterizing when two nodes in a document cannot be distinguished by an expression in the fragment under consideration. It turns out for those fragments we consider that this notion of expression equivalence of nodes is equivalent to an appropriate generalization of the classic notion of bisimilarity [25]. The second step of our methodology then consists of bootstrapping this result to a characterization for when a binary relation on the nodes of a given document is definable by an expression in the fragment (in the sense of the previous paragraph).

We refer to this perspective on the semantics of XPath at the document level as the "global view." In contrast with this global view, there is also a "local view" which we consider. In this view, one is only interested in the nodes to which one can navigate starting from a particular given node in the document under consideration. From this perspective, a set of nodes of that document can be seen as the end points of a set of paths starting at the given node. For each of the XPath fragments considered, we characterize when such a set represents the set of *all* paths starting at the given node defined by some expression in the

fragment. These characterizations are derived from the corresponding characterizations in the "global view," and turn out to be particularly elegant in the important special case where the starting node is the root.

In this paper, we study several natural XPath fragments. The most expressive among them is the *XPath algebra* which permits the self, parent, and child operators, predicates, compositions, and the boolean operators union, intersection, and difference. (Since we work at the document level, i.e., the document is given, there is no need to include the ancestor and descendant operators as primitives.) We also consider the *core XPath algebra*, which is the XPath algebra without intersection and difference at the expression level. The core XPath algebra is the adaptation to our setting of Core XPath of Gottlob et al. [16, 17, 26]. Of both of these algebras, we also consider various "downward" and "upward" fragments without the parent and child operator, respectively. We also study "positive" variants of all the fragments considered, without the difference operator.

Our strategy is to introduce and characterize generalizations of each of these practical fragments, towards a broader perspective on relation algebras on trees. These generalizations are based on a simple notion of path counting, a feature which also appears in XPath.

The robustness of the characterizations provided in this paper is further strengthened by their feasibility. As discussed in Section 9, the global and local definability problems for each of the XPath fragments are decidable in polynomial time. This feasibility hints towards efficient partitioning and reduction techniques on both the set of nodes and the set of paths in a document. Such techniques may fruitfully applied towards, e.g., document compression [27], access control [28], and designing indexes for query processing [12, 29, 30].

We proceed in the paper as follows. In Section 2, we formally define documents and the algebras, and then in Section 3, we define a notion of "signatures" which will be essential in the sequel. In Section 4, we define the semantic and syntactic notions of node distinguishability necessary to obtain our desired structural characterizations. In the balance of the paper, we apply our two-step methodology to link semantic expression equivalence in the languages to appropriate structural syntactic equivalence notions. In particular, we give structural characterizations, under both the global and local views,

- of "strictly" (Section 5) and "weakly" (Section 6) downward languages, and their positive variants;
- of upward languages and their positive variants (Section 7); and,
- of languages with both downward and upward navigation, and their positive variants (Section 8).

Along the way, we also establish the equivalence of some of these fragments, using the structural characterizations obtained. We conclude in Section 9 with a discussion of some ramifications of our results and directions for further study.

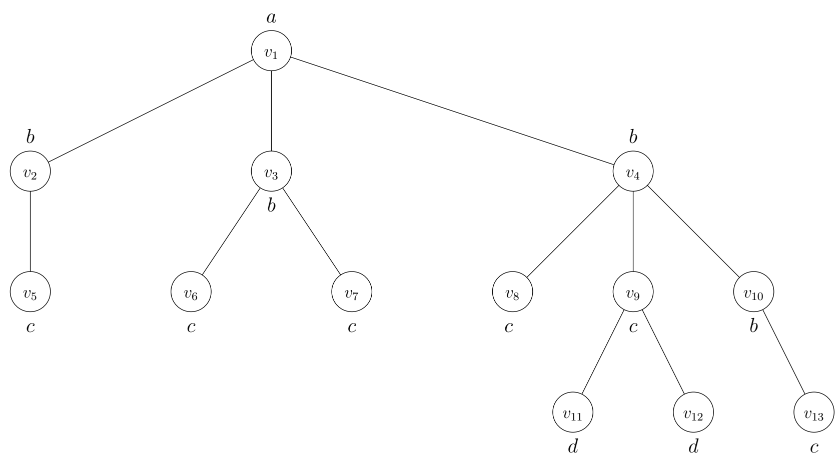

Figure 1: Example document.

2. Documents and navigation

In this paper, we are interested in navigating over documents in the form of unordered labeled trees. Formally, we denote such a document as $D = (V, Ed, r, \lambda)$, with D the document name, V the set of nodes of the tree, Ed the set of edges of the tree, r the root of the tree, and $\lambda : V \to \mathcal{L}$ a function assigning to each node a label from some infinite set of labels $\mathcal{L}$.

Example 2.1. Figure 1 shows an example of a document that will be used throughout the paper. Here, $r = v_1$ is the root of the tree with label $\lambda(v_1) = a$.

We next define a set of operations on documents, as tabulated in Table 1. The left column shows the syntax of the operation, and the right column its semantics, given a document $D = (V, Ed, r, \lambda)$. Notice that, in each case, the result is a binary relation on the nodes of the document.

The *basic algebra*, denoted $\mathcal{X}$, is the language consisting of all expressions built from $\emptyset$, ε, $\hat{\ell}$ with $\ell \in \mathcal{L}$, composition ("/"), and union ("$\cup$"). The basic algebra $\mathcal{X}$ can be extended by adding some of the other operations in Table 1, which we call *nonbasic*. If E is a set of nonbasic operations, then $\mathcal{X}(E)$ denotes the algebra obtained by adding the operations in E to the basic algebra $\mathcal{X}$. When writing expressions, we assume that unary operations take precedence over binary operations, and that composition takes precedence over the set operations.

Notice that we do not consider transitive closure operations such as the descendant ("$\downarrow^*$") or ancestor ("$\uparrow^*$") operations of XPath. The reason for this is that, in this paper, we only consider navigation within a single, given document.

Example 2.2. Consider the document D in Figure 1. Let e be the expression $\uparrow/\pi_1(\downarrow/\hat{b}/\downarrow/\hat{c}) - \mathrm{ch}_{\geq 2}(\varepsilon)/\uparrow$ in the language $\mathcal{X}(\downarrow, \uparrow, \pi_1, \mathrm{ch}_{\geq 2}, -)$ (or, for that

Table 1: Binary operations on documents. The left column shows the syntax of the operation, and the right column its semantics, given a document $D = (V, Ed, r, \lambda)$. Below, ℓ is a label in $\mathcal{L}$ and $k \geq 1$ a natural number. Furthermore, in the recursive definitions, e, e_1, and e_2 represents expressions built with the operations.

Syntax	Semantics
$\emptyset$	$\emptyset(D) = \emptyset$
ε	$\varepsilon(D) = \{(v, v) \mid v \in V\}$
$\hat{\ell}$	$\hat{\ell}(D) = \{(v, v) \mid v \in V \;\&\; \lambda(v) = \ell\}$
$\downarrow$	$\downarrow(D) = Ed$
$\uparrow$	$\uparrow(D) = Ed^{-1}$
$\pi_1(e)$	$\pi_1(e)(D) = \{(v, v) \mid (\exists w)(v, w) \in e(D)\}$
$\pi_2(e)$	$\pi_2(e)(D) = \{(w, w) \mid (\exists v)(v, w) \in e(D)\}$
e^{-1}	$e^{-1}(D) = e(D)^{-1}$
$\mathrm{ch}_{\geq k}(e)$	$\mathrm{ch}_{\geq k}(e)(D) = \{(v, v) \mid v \in V \;\&\; \lvert\{w \mid (v, w) \in Ed \;\&\; (w, w) \in \pi_1(e)(D)\rvert \geq k\}$
e_1/e_2	$e_1/e_2(D) = \{(u, w) \mid (\exists v)((u, v) \in e_1(D) \;\&\; (v, w) \in e_2(D))\}$
$e_1 \cup e_2$	$e_1 \cup e_2(D) = e_1(D) \cup e_2(D)$
$e_1 \cap e_2$	$e_1 \cap e_2(D) = e_1(D) \cap e_2(D)$
$e_1 - e_2$	$e_1 - e_2(D) = e_1(D) - e_2(D)$

matter, in any language $\mathcal{X}(E)$ with $\{\downarrow, \uparrow, \pi_1, \mathrm{ch}_{\geq 2}, -\} \subseteq E$). Then, $e(D) = \{(v_2, v_1), (v_8, v_4), (v_{10}, v_4)\}$.

Not all the above operations are primitive, however. For instance, intersection ("$\cap$") is expressible as soon a set difference ("$-$") is expressible, since, for any two sets A and B, $A \cap B = A - (A - B)$. Even more eliminations are possible in the following setting.

Proposition 2.3. *Let E be a set of nonbasic operations containing set difference "$-$") or intersection ("$\cap$") for which "$\downarrow$" and "$\uparrow$" are both contained in E or both not contained in E. Then, for each expression e in $\mathcal{X}(E)$, there is an equivalent expression in $\mathcal{X}(E - \{\pi_1, \pi_2, {}^{-1}\})$.*

Proof. First, we eliminate both projections using the identities

$$\begin{aligned} \pi_1(e) &= (e/e^{-1}) \cap \varepsilon; \\ \pi_2(e) &= (e^{-1}/e) \cap \varepsilon. \end{aligned}$$

Hence, each expression in $\mathcal{X}(E)$ can be replaced by an equivalent expression in $\mathcal{X}((E \cup \{{}^{-1}\}) - \{\pi_1, \pi_2\})$. It remains to show that we can eliminate inverse ("${}^{-1}$"). This follows from the following identities. In these, $D = (V, Ed, r, \lambda)$ is a document, $\ell \in \mathcal{L}$ is a label, $k \geq 1$ is a natural number, and e, e_1 and e_2 are expressions in $\mathcal{X}(E)$.

- $\emptyset^{-1}(D) = \emptyset(D)$;
- $\varepsilon^{-1}(D) = \varepsilon(D)$;

- $\hat{\ell}^{-1}(D) = \hat{\ell}(D)$;
- $\downarrow^{-1}(D) = \uparrow(D)$;
- $\uparrow^{-1}(D) = \downarrow(D)$;
- $(e^{-1})^{-1}(D) = e(D)$;
- $(e_1/e_2)^{-1}(D) = e_2^{-1}/e_1^{-1}(D)$;
- $\mathrm{ch}_{\geq k}(e)^{-1}(D) = \mathrm{ch}_{\geq k}(e)(D)$;
- $(e_1 \cup e_2)^{-1}(D) = e_1^{-1} \cup e_2^{-1}(D)$;
- $(e_1 \cap e_2)^{-1}(D) = e_1^{-1} \cap e_2^{-1}(D)$;
- $(e_1 - e_2)^{-1}(D) = e_1^{-1} - e_2^{-1}(D)$.

□

Notice that in a language with both upward ("$\uparrow$") and downward ("$\downarrow$") navigation, the identities $\pi_1(e)(D) = \pi_2(e^{-1})(D)$ and $\pi_2(e)(D) = \pi_1(e^{-1})(D)$ imply that one projection operation can be eliminated in favor of the other. Hence, it does not make sense to consider the projection operations separately.

Some counting operations ("$\mathrm{ch}_{\geq k}(e)$") can also be simulated. One can easily verify the following.

Proposition 2.4. *Let $D = (V, Ed, r, \lambda)$ be a document. Then,*

1. $ch_{\geq 1}(e)(D) = \pi_1(\downarrow/e)(D)$;
2. $ch_{\geq 2}(e)(D) = \pi_1(\downarrow/(\pi_1(e)/\uparrow/\downarrow/\pi_1(e) - \varepsilon))(D)$; *and*
3. $ch_{\geq 3}(e)(D) = \pi_1(\downarrow/((\pi_1(e)/\uparrow/\downarrow/\pi_1(e) - \varepsilon)/(\pi_1(e)/\uparrow/\downarrow/\pi_1(e) - \varepsilon) - \varepsilon)(D)$

Example 2.5. Consider again the expression $e := \uparrow/\pi_1(\downarrow/\hat{b}/\downarrow/\hat{c}) - \mathrm{ch}_{\geq 2}(\varepsilon)/\uparrow$ of Example 2.2. Using Proposition 2.4, and making some straightforward simplifications, we can rewrite e as $\uparrow/\pi_1(\downarrow/\hat{b}/\downarrow/\hat{c}) - \pi_1(\downarrow/(\uparrow/\downarrow - \varepsilon))/\uparrow$, an expression of $\mathcal{X}(\downarrow, \uparrow, \pi_1, -)$. Alternatively, one can use Proposition 2.3 and the techniques exhibited in its proof to rewrite e as

$$\uparrow/(\downarrow/\hat{b}/\downarrow/\hat{c}/\uparrow/\downarrow) - \downarrow/((\uparrow/\downarrow - \varepsilon)/(\uparrow/\downarrow - \varepsilon) \cap \varepsilon)/\uparrow,$$

an expression in $\mathcal{X}(\downarrow, \uparrow, \cap, -)$. Finally, we invite the reader to verify that e can also be rewritten as

$$\pi_1(\varepsilon - \pi_1(\downarrow/(\uparrow/\downarrow - \varepsilon)))/\uparrow/\pi_1(\downarrow/\hat{b}/\downarrow/\hat{c}),$$

also an expression of $\mathcal{X}(\downarrow, \uparrow, \pi_1, -)$.

We shall call the language $\mathcal{X}(\downarrow, \uparrow, \pi_1, \pi_2, .^{-1}, \cap, -)$, which by Proposition 2.3 is equivalent to $\mathcal{X}(\downarrow, \uparrow, -)$, the *XPath algebra.*[1] This is justified by the following result.

Proposition 2.6. *Given a single document $D = (V, Ed, r, \lambda)$, the XPath algebra is equivalent to XPath.*

Proof. Notice that ℓ in XPath [14] is simulated by $\downarrow/\hat{\ell}$ in the XPath algebra. Furthermore, $\hat{\ell}$ in the XPath algebra is simulated by $\varepsilon[\text{label} = \ell]$ in XPath. The proof is complete if, for each predicate P in XPath, there exists an XPath algebra expression e such that $e(D) = \{(n, n) \mid n \in P(D)\}$. This is proved by structural induction:

1. if P is an XPath expression without predicates, then take $e := \pi_1(f)$, with f the XPath algebra expression obtained from P by replacing everywhere ℓ by $\downarrow/\hat{\ell}$.
2. if P is label $= \ell$, then take $e := \hat{\ell}$.
3. if P is $\neg Q$, with Q an XPath predicate, then take $e := \varepsilon - f$, with f the XPath algebra expression corresponding to Q.
4. if P is $Q_1 \wedge Q_2$, with Q_1 and Q_2 XPath predicates, then take $e := f_1 \cap f_2$, with f_1 and f_2 the XPath algebra expressions corresponding to Q_1 and Q_2, respectively.
5. if P is $Q_1 \vee Q_2$, with Q_1 and Q_2 XPath predicates, then take $e := f_1 \cup f_2$, with f_1 and f_2 the XPath algebra expressions corresponding to Q_1 and Q_2, respectively.

$\square$

Besides the *standard languages* $\mathcal{X}(E)$, with E a set of nonbasic operations, we also consider the so-called *core languages* $\mathcal{C}(E)$. More concretely, $\mathcal{C}(E)$ is defined recursively in the same way as $\mathcal{X}(E - \{\cap, -\})$, except that in expressions of the form $\pi_1(f)$, and $\pi_2(f)$, f may be a boolean combination of expressions of the language using union and the operations in $E \cap \{\cap, -\}$, rather than just an expression of the language.

The above terminology is inspired by the fact that $\mathcal{C}(\downarrow, \uparrow, \pi_1, \pi_2, -, \cap)$, the language which we call the *core XPath algebra*, is the adaptation to our setting of Core XPath of Gottlob and Koch [16].

Example 2.7. Continuing with Example 2.5, we consider again the expression $e := \uparrow/\pi_1(\downarrow/\hat{b}/\downarrow/\hat{c}) - \text{ch}_{\geq 2}(\varepsilon)/\uparrow$ of Example 2.2. Obviously, there is no core language of which e is an expression, as set difference ("$-$") occurs at the outer level, and not in a subexpression f which in turn is embedded in a subexpression of the form $\pi_1(f)$ or $\pi_2(f)$. However, in Example 2.5, the expression e has been shown to be equivalent to

$$\pi_1(\varepsilon - \pi_1(\downarrow/(\uparrow/\downarrow - \varepsilon)))/\uparrow/\pi_1(\downarrow/\hat{b}/\downarrow/\hat{c}),$$

[1] Note that the XPath algebra corresponds to the (full) relation algebra of Tarski [2], adapted to our setting (cf. [8]).

Table 2: Languages studied in this paper.

Language	*Relation algebra fragment*
strictly downward (core) XPath algebra with counting up to k	$\mathcal{X}(\downarrow, \pi_1, \mathrm{ch}_{\geq 1}(.), \ldots, \mathrm{ch}_{\geq k}(.), -) = \mathcal{C}(\downarrow, \pi_1, \mathrm{ch}_{\geq 1}(.), \ldots, \mathrm{ch}_{\geq k}(.), -)$
strictly downward (core) positive XPath algebra	$\mathcal{X}(\downarrow, \pi_1, \cap) = \mathcal{C}(\downarrow, \pi_1, \cap)$
weakly downward (core) XPath algebra with counting up to k	$\mathcal{X}(\downarrow, \pi_1, \pi_2, \mathrm{ch}_{\geq 1}(.), \ldots, \mathrm{ch}_{\geq k}(.), -) = \mathcal{C}(\downarrow, \pi_1, \pi_2, \mathrm{ch}_{\geq 1}(.), \ldots, \mathrm{ch}_{\geq k}(.), -)$
weakly downward (core) positive XPath algebra	$\mathcal{X}(\downarrow, \pi_1, \pi_2) = \mathcal{X}(\downarrow, \pi_1, \pi_2, \cap) = \mathcal{C}(\downarrow, \pi_1, \pi_2, \cap)$
strictly upward (core) XPath algebra	$\mathcal{X}(\uparrow, \pi_1, -) = \mathcal{C}(\uparrow, \pi_1, -)$
strictly upward (core) positive XPath algebra	$\mathcal{X}(\uparrow, \pi_1, \cap) = \mathcal{C}(\uparrow, \pi_1, \cap)$
weakly upward languages	*see Section 7.2*
XPath algebra	$\mathcal{X}(\downarrow, \uparrow, \pi_1, \pi_2, .^{-1}, \cap, -) = \mathcal{X}(\downarrow, \uparrow, -)$
XPath algebra with counting up to k	$\mathcal{X}(\downarrow, \uparrow, \mathrm{ch}_{\geq 1}(.), \ldots, \mathrm{ch}_{\geq k}(.), -)$
core XPath algebra	$\mathcal{C}(\downarrow, \uparrow, \pi_1, \pi_2, -, \cap)$
core XPath algebra with counting up to k	$\mathcal{C}(\downarrow, \uparrow, \pi_1, \pi_2, \mathrm{ch}_{\geq 1}(.), \ldots, \mathrm{ch}_{\geq k}(.), -)$
(core) positive XPath algebra ([31])	$\mathcal{X}(\downarrow, \uparrow, \cap) = \mathcal{X}(\downarrow, \uparrow, \pi_1, \pi_2) = \mathcal{C}(\downarrow, \uparrow, \pi_1, \pi_2, \cap)$

which is an expression of $\mathcal{C}(\downarrow, \uparrow, \pi_1, -, \cap)$, and hence also of the core XPath algebra.

Given a set of nonbasic operators E, an expression in $\mathcal{X}(E)$ can in general *not* be converted to an equivalent expression in $\mathcal{C}(E)$, however, as will follow from the results of this paper, even though there are exceptions (Section 5, Theorem 5.19).

Table 2 gives an overview of the various relation algebra fragments we investigate below.

To conclude this section, we observe that, given a document and an expression, we have defined the semantics of that expression as a binary relation over the nodes of the document, i.e., as a set of pair of nodes. From the perspective of navigation, however, it is useful to be able to say that an expression allows one to navigate from one node of the document to another. For this purpose, we introduce the following notation.

Definition 2.8. Let e be an arbitrary expression, and let $D = (V, Ed, r, \lambda)$ be a document. For $v \in V$, $e(D)(v) := \{w \mid (v, w) \in e(D)\}$.

Definition 2.8 reflects the "local" perspective of an expression working on particular nodes of a document, rather than the "global" perspective of working on an entire document.

Example 2.9. Consider again the expression $e := \uparrow/\pi_1(\downarrow/\hat{b}/\downarrow/\hat{c}) - \mathrm{ch}_{\geq 2}(\varepsilon)/\uparrow$ of Example 2.2. We have established that, for the document D in Figure 1, $e(D) = \{(v_2, v_1), (v_8, v_4), (v_{10}, v_4)\}$. Hence, $e(D)(v_8) = \{v_4\}$ and $e(D)(v_1) = \emptyset$.

3. Signatures

Given a pair of nodes in a document, there is a unique path in that document (not taking into account the direction of the edges) to navigate from the first to the second node, in general by going a few steps upward in the tree, and then going a few steps downward. We call this the *signature* of that pair of nodes, and shall formally represent it by an expression in $\mathcal{X}(\downarrow, \uparrow)$.

Definition 3.1. Let $D = (V, Ed, r, \lambda)$ be a document, and let $v, w \in V$. The *signature* of the pair (v, w), denoted $\mathrm{sig}(v, w)$, is the expression in $\mathcal{X}(\downarrow, \uparrow)$ that is recursively defined, as follows:

- if $v = w$, then $\mathrm{sig}(v, w) := \varepsilon$;
- if v is an ancestor of w, and z is the child of v on the path from v to w, then $\mathrm{sig}(v, w) := \downarrow/\mathrm{sig}(z, w)$;
- otherwise[2], if z is the parent of v, then $\mathrm{sig}(v, w) := \uparrow/\mathrm{sig}(z, w)$.

Given nodes v and w of a document $D = (V, Ed, r, \lambda)$, we denote by $\mathrm{top}(v, w)$ the unique node on the undirected path from v to w that is an ancestor of both v and w. Clearly,

$$\mathrm{sig}(v, w) = \mathrm{sig}(v, \mathrm{top}(v, w))/\mathrm{sig}(\mathrm{top}(v, w), w) = \uparrow^m/\downarrow^n,$$

where m, respectively n, is the distance from $\mathrm{top}(v, w)$ to v, respectively w; and, for an expression e and a natural number $i \geq 1$, e^i denotes the i-fold composition of e.[3] (We put $e^0 := \varepsilon$.)

The signature of a pair of nodes of a document can be seen as a description of the unique path connecting these nodes, but also as an expression that can be applied to the document under consideration. We shall often exploit this duality.

[2]In particular, $v \neq r$.

[3]Here, and elsewhere in this paper, equality between expressions must be interpreted at the semantic and not at the syntactic level, i.e., for two expressions e_1 and e_2 in one of the languages considered here, $e_1 = e_2$ means that, for each document D, $e_1(D) = e_2(D)$.

Example 3.2. For the document D in Figure 1, $\mathrm{sig}(v_1, v_1) = \varepsilon$, $\mathrm{sig}(v_1, v_2) = \downarrow$, $\mathrm{sig}(v_6, v_4) = \uparrow^2/\downarrow$, and $\mathrm{sig}(v_{11}, v_5) = \uparrow^3/\downarrow^2$. We have that

$$\begin{aligned}\mathrm{sig}(v_{11}, v_5)(D) \quad = \quad & \{(v_{11}, v_5), (v_{12}, v_5), (v_{13}, v_5), (v_{11}, v_6), (v_{12}, v_6), (v_{13}, v_6),\\ & (v_{11}, v_7), (v_{12}, v_7), (v_{13}, v_7), (v_{11}, v_8), (v_{12}, v_8), (v_{13}, v_8),\\ & (v_{11}, v_9), (v_{12}, v_9), (v_{13}, v_9), (v_{11}, v_{10}), (v_{12}, v_{10}), (v_{13}, v_{10})\}.\end{aligned}$$

Notice that not each pair in the result has the same signature as (v_{11}, v_5). For instance, $\mathrm{sig}(v_{11}, v_8) = \uparrow^2/\downarrow$ and $\mathrm{sig}(v_{11}, v_9) = \uparrow$.

Now, let (v_1, w_1) and (v_2, w_2) be two pairs of nodes in a document $D = (V, Ed, r, \lambda)$. We say that (v_1, w_1) *subsumes* (v_2, w_2), denoted $(v_1, w_1) \gtrsim (v_2, w_2)$, if (v_2, w_2) is in $\mathrm{sig}(v_1, w_1)(D)$. We say that (v_1, w_1) are (v_2, w_2) *congruent*, denoted $(v_1, w_1) \cong (v_2, w_2)$, if $(v_1, w_1) \gtrsim (v_2, w_2)$ and $(v_2, w_2) \gtrsim (v_1, w_1)$. It can be easily seen that, in this case, $\mathrm{sig}(v_1, w_1) = \mathrm{sig}(v_2, w_2)$. Informally speaking, the path from v_1 to w_1 has then the same shape as the path from v_2 to w_2.

Example 3.3. Consider again Example 3.2. Clearly, (v_{11}, v_5) subsumes each pair of nodes in $\mathrm{sig}(v_{11}, v_5)(D)$, e.g., $(v_{11}, v_5) \gtrsim (v_{12}, v_6)$ and $(v_{11}, v_5) \gtrsim (v_{12}, v_9)$. Notice that also $(v_{12}, v_6) \gtrsim (v_{11}, v_5)$, and hence $(v_{11}, v_5) \cong (v_{12}, v_6)$. However, $(v_{12}, v_9) \not\gtrsim (v_{11}, v_5)$. Hence, these pairs are not congruent.

By definition, subsumption is captured by the "sig" expression. One may wonder if there also exists an expression that precisely captures congruence. This is the case in the following situations.

Proposition 3.4. *Let $D = (V, Ed, r, \lambda)$ be a document and let $v_1, v_2, w_1, w_2 \in V$. Then,*

1. *if v_1 is an ancestor of w_1 or vice versa, $(v_1, w_1) \cong (v_2, w_2)$ if and only if $(v_2, w_2) \in sig(v_1, w_1)(D)$;*
2. *otherwise, let $sig(v_1, w_1) = \uparrow^m/\downarrow^n$. Then, as $m \geq 1$ and $n \geq 1$, $(v_1, w_1) \cong (v_2, w_2)$ if and only if $(v_2, w_2) \in \uparrow^m/\downarrow^n - \uparrow^{m-1}/\downarrow^{n-1}(D)$.*

Proof.
1. As the "only if" is trivial, it suffices to consider the "if," which follows from a straightforward induction argument.
2. As the "only if" is straightforward, we only consider the "if." Let $t_2 := \uparrow^m(D)(v_2)$. Since $w_2 \in \downarrow^n(D)(t_2)$, t_2 is a common ancestor. Let v_2' and w_2' be the children of t_2 on the path to v_2 and w_2, respectively. If $v_2' = w_2'$, then $(v_2, w_2) \in \uparrow^{m-1}/\downarrow^{n-1}(D)$, a contradiction, Hence, $v_2' \neq w_2'$ and $t_2 = \mathrm{top}(v_2, w_2)$, and $\mathrm{sig}(v_2, w_2) = \uparrow^m/\downarrow^n = \mathrm{sig}(v_1, w_1)$.

□

For later use, but also because of their independent interest, we finally note the following fundamental properties of subsumption and congruence.

Proposition 3.5. *Let v, w, v_1, w_1, z_1, v_2, w_2, and z_2 be nodes of a document D. Then the following properties hold.*

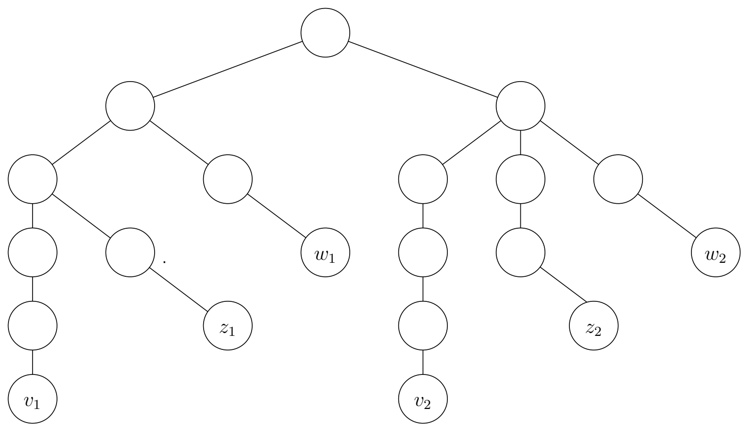

Figure 2: Document of Example 3.6.

1. $(v, v) \gtrsim (w, w)$.
2. $(v_1, w_1) \gtrsim (v_2, w_2)$ *implies that* $(w_1, v_1) \gtrsim (w_2, v_2)$.
3. *If top*(v_1, z_1) *is also an ancestor of* w_1, *then* $(v_1, w_1) \gtrsim (v_2, w_2)$ *and* $(w_1, z_1) \gtrsim (w_2, z_2)$ *imply that* $(v_1, z_1) \gtrsim (v_2, z_2)$.
4. *All properties above also hold when subsumption is replaced by congruence, provided that, in item 3, the condition "top*(v_2, z_2) *is also an ancestor of* w_2*" is added.*

Proof. All properties are straightforward, except for Property 3. So, assume that $(v_1, w_1) \gtrsim (v_2, w_2)$ and $(v_1, z_1) \gtrsim (v_2, z_2)$. Hence, $(v_2, w_2) \in \mathrm{sig}(v_1, w_1)(D)$ and $(v_2, z_2) \in \mathrm{sig}(v_1, z_1)(D)$, as a consequence of which

$$(v_2, z_2) \in \mathrm{sig}(v_1, w_1)/\mathrm{sig}(w_1, z_1)(D).$$

For the sake of abbreviation, let $t_1 := \mathrm{top}(v_1, w_1)$ and $u_1 := \mathrm{top}(w_1, z_1)$. Using these nodes, we can write

$$\mathrm{sig}(v_1, w_1)/\mathrm{sig}(w_1, z_1) = \mathrm{sig}(v_1, t_1)/\mathrm{sig}(t_1, w_1)/\mathrm{sig}(w_1, u_1)/\mathrm{sig}(u_1, z_1),$$

which is equal to $\mathrm{sig}(v_1, s_1)/\mathrm{sig}(s_1, z_1)$, where s_1 is the higher of t_1 and u_1 in D. Notice that s_1 is a common ancestor of v_1 and z_1, as a consequence of which it is also an ancestor of $\mathrm{top}(v_1, z_1)$, the least common ancestor of v_1 and z_1. By assumption, $\mathrm{top}(v_1, z_1)$ is a common ancestor of v_1, w_1, and z_1, and hence also of $\mathrm{top}(v_1, w_1)$ and $\mathrm{top}(w_1, z_1)$, the highest of which is s_1. Thus, $s_1 = \mathrm{top}(v_1, z_1)$, and, therefore, $\mathrm{sig}(v_1, s_1)/\mathrm{sig}(s_1, z_1) = \mathrm{sig}(v_1, z_1)$. In summary, $(v_2, z_2) \in \mathrm{sig}(v_1, z_1)(D)$, and hence $(v_1, z_1) \gtrsim (v_2, z_2)$. □

Observe that the condition in Proposition 3.5, (3), is necessary for that part of the proposition to hold, as shown by the following counterexample.

Example 3.6. Consider the document in Figure 2. Labels have been omitted, because they are not relevant in this discussion. (We assume all nodes have the same label.) Observe that $(v_1, w_1) \cong (v_2, w_2)$ and $(w_1, z_1) \cong (w_2, z_2)$. However, $\mathrm{top}(v_1, z_1)$ is *not* an ancestor of w_1, hence, Proposition 3.5, (3), is *not* applicable. We see that, indeed, (v_1, z_1) does *not* subsume (v_2, z_2), let alone that (v_2, z_2) and (v_2, w_2) would be congruent.

4. Distinguishability of nodes in a document

We wish to link the distinguishing power of a navigational language on a document to syntactic conditions which can readily be verified on that document. As argued before, the action of an expression on a document can be interpreted as (1) returning pairs of nodes, or (2) given a node, returning the set of nodes that can be reached from that node. We shall refer to the first interpretation as the *pairs semantics*, and to the second interpretation as the *node semantics*. In this section, we propose suitable semantic and syntactic notions of distinguishability for the node semantics.

4.1. Distinguishability of nodes at the semantic level

We propose the following distinguishability criterion based on the emptiness or nonemptiness of the set of nodes that can be reached by applying an arbitrary expression of the language under consideration.

Definition 4.1. Let L be one of the languages considered in Section 2. Let $D = (V, Ed, r, \lambda)$ be a document, and let $v_1, v_2 \in V$. Then,

1. v_1 and v_2 are *expression-related*, denoted $v_1 \geq_{\mathrm{exp}} v_2$, if, for each expression e in L, $e(D)(v_1) \neq \emptyset$ implies $e(D)(v_2) \neq \emptyset$; and
2. v_1 and v_2 are *expression-equivalent*, denoted $v_1 \equiv_{\mathrm{exp}} v_2$, if $v_1 \geq_{\mathrm{exp}} v_2$ and $v_2 \geq_{\mathrm{exp}} v_1$.

In principle, we should have reflected the language under consideration in the notation for expression-equivalence. As the language under consideration will always be clear from the context, we chose not to do so in order to avoid overloaded notation.

The following observation is useful.

Proposition 4.2. *Let E be a set of nonbasic operations containing first projection ("π_1") and set difference ("$-$"). Consider expression-equivalence with respect to $\mathcal{X}(E)$. Let $D = (V, Ed, r, \lambda)$ be a document, and let $v_1, v_2 \in V$. Then, $v_1 \equiv_{\mathrm{exp}} v_2$ if and only if $v_1 \geq_{\mathrm{exp}} v_2$.*

Proof. Assume that $v_2 \not\geq_{\mathrm{exp}} v_1$. Then there exists an expression f in $\mathcal{X}(E)$ such that $f(D)(v_2) \neq \emptyset$ and $f(D)(v_1) = \emptyset$. Now consider $e := \pi_1(\varepsilon - \pi_1(f))$. Clearly, $e(D)(v_2) = \emptyset$ and $e(D)(v_1) \neq \emptyset$, hence $v_1 \not\geq_{\mathrm{exp}} v_2$. By contraposition, $v_1 \geq_{\mathrm{exp}} v_2$ implies $v_2 \geq_{\mathrm{exp}} v_1$, and hence also $v_1 \equiv_{\mathrm{exp}} v_2$. □

4.2. Distinguishability of nodes at the syntactic level

Our syntactic criterion of distinguishability is based on the similarity of the documents locally around the nodes under consideration. In order to decide this similarity, we shall consider a hierarchy for the degree of coarseness by which we compare the environments of those nodes. We shall also consider variants for the cases where from the given nodes of the document we (1) only look downward; (2) only look upward; or (3) look in both directions.

4.2.1. Downward distinguishability

For the downward case, we consider the following syntactic notions of distinguishability of nodes. They are all defined recursively on the height of the first node.

Definition 4.3. Let $D = (V, Ed, r, \lambda)$ be a document, let $v_1, v_2 \in V$, and let $k \geq 1$. Then, v_1 and v_2 are *downward-k-equivalent*, denoted $v_1 \equiv_{\downarrow}^{k} v_2$, if

1. $\lambda(v_1) = \lambda(v_2)$;
2. for each child w_1 of v_1, there exists a child w_2 of v_2 such that $w_1 \equiv_{\downarrow}^{k} w_2$, and vice versa;
3. for each child w_1 of v_1 and w_2 of v_2 such that $w_1 \equiv_{\downarrow}^{k} w_2$, $\min(|\bar{w}_1|, k) = \min(|\bar{w}_2|, k)$, where, for $i = 1, 2$, $\bar{w}_i$ is the set of all siblings of w_i (including w_i itself) that are downward k-equivalent to w_i.[4]

For $k = 1$, the third condition in the above definition is trivially satisfied. In the literature, downward 1-equivalence is usually referred to as *bisimilarity* [25].

Example 4.4. Consider again the example document in Figure 1. Notice that $v_2 \equiv_{\downarrow}^{k} v_{10}$ for any value of $k \geq 1$. We also have that $v_2 \equiv_{\downarrow}^{1} v_3$, and, for any value of $k \geq 2$, $v_2 \not\equiv_{\downarrow}^{k} v_3$. Finally, notice that $v_3 \not\equiv_{\downarrow}^{k} v_4$ for any value of $k \geq 1$.

The following is immediate from the second condition in the Definition 4.3.

Proposition 4.5. *Let $D = (V, Ed, r, \lambda)$ be a document, let $v_1, v_2 \in V$, and let $k \geq 1$. If $v_1 \equiv_{\downarrow}^{k} v_2$, then v_1 and v_2 have equal height*[5] *in D.*

The following property of downward-k-equivalence will turn out to be very useful in the sequel.

Proposition 4.6. *Let $D = (V, Ed, r, \lambda)$ be a document, and let $k \geq 1$. Let "$\equiv$" be an equivalence relation on V such that, for all $v_1, v_2 \in V$ with $v_1 \equiv v_2$,*

1. *$\lambda(v_1) = \lambda(v_2)$;*
2. *for each child w_1 of v_1, there exists a child w_2 of v_2 such that $w_1 \equiv w_2$, and vice versa; and*

---

[4]For a set A, $|A|$ denotes the cardinality of A.

[5]By the height of a node, we mean the length of the longest path from that node to a leaf.

3. *for each child w_1 of v_1 and each child w_2 of v_2 such that $w_1 \equiv w_2$, $\min(|\tilde{w}_1|, k) = \min(|\tilde{w}_2|, k)$, where, for $i = 1, 2$, $\tilde{w}_i$ is the set of all siblings of v_i (including v_i itself) that are equivalent to v_i under "$\equiv$."*

Then, for all $v_1, v_2 \in V$, $v_1 \equiv v_2$ implies $v_1 \equiv_{\downarrow}^{k} v_2$.

Proof. By induction of the height of v_1.

If v_1 is a leaf, the second condition above implies that v_2 must also be a leaf. By the first condition, $\lambda(v_1) = \lambda(v_2)$. Hence, $v_1 \equiv_{\downarrow}^{k} v_2$.

If v_1 is not a leaf, we still have, by the first condition, that $\lambda(v_1) = \lambda(v_2)$. Hence the first condition in the definition of $v_1 \equiv_{\downarrow}^{k} v_2$ (Definition 4.3) is satisfied.

The second condition in the definition of $v_1 \equiv_{\downarrow}^{k} v_2$ follows from the second condition above and the induction hypothesis.

It remains to show that also the third condition in the definition of $v_1 \equiv_{\downarrow}^{k} v_2$ holds. Thereto, let w_1 be a child of v_1 and w_2 be a child of v_2 such that $w_1 \equiv_{\downarrow}^{k} w_2$. We show that $\min(|\bar{w}_1|, k) = \min(|\bar{w}_2|, k)$, where, for $i = 1, 2$, $\bar{w}_i$ is the set of all siblings of w_i (including w_i itself) that are downward k-equivalent to w_i. Let $\{W_{11}, \ldots, W_{1\ell}\}$ be the coarsest partition of $\bar{w}_1$ in $\equiv$-equivalent nodes, and let $\{W_{21}, \ldots, W_{2\ell}\}$ be the coarsest partition of $\bar{w}_2$ in $\equiv$-equivalent nodes. By the induction hypothesis and the second condition above, both partitions have indeed the same size. It follows furthermore that no node of $\bar{w}_1$ is $\equiv$-equivalent with a child of v_1 outside $\bar{w}_1$, and that no node of $\bar{w}_2$ is $\equiv$-equivalent with a child of v_2 outside $\bar{w}_2$. Without loss of generality, we may assume that, for $i = 1, \ldots, \ell$, every node in W_{1i} is $\equiv$-equivalent to every node in W_{2i}. Hence, by the third condition above, $\min(|W_{1i}|, k) = \min(|W_{2i}|, k)$. We now distinguish two cases.

1. For all $i = 1, \ldots, \ell$, $|W_{1i}| < k$. Then, for all $i = 1, \ldots, \ell$, $|W_{1i}| = |W_{2i}|$. It follows that $|\bar{w}_1| = |\bar{w}_2|$, and, hence, also that $\min(|\bar{w}_1|, k) = \min(|\bar{w}_2|, k)$.
2. For some i, $1 \leq i \leq \ell$, $|W_{1i}| \geq k$. Then, $|W_{2i}| = |W_{1i}| \geq k$. Hence, $|\bar{w}_1| \geq k$ and $|\bar{w}_2| \geq k$. It follows that $\min(|\bar{w}_1|, k) = \min(|\bar{w}_2|, k) = k$.

We conclude that, in both cases, the third condition in the definition of $v_1 \equiv_{\downarrow}^{k} v_2$ is also satisfied. □

So, given a document $D = (V, Ed, r, \lambda)$, downward-k-equivalence is the coarsest equivalence relation on V satisfying Proposition 4.6.

A straightforward application of Proposition 4.6 yields

Corollary 4.7. *Let $k \geq 1$. Let $D = (V, Ed, r, \lambda)$ be a document, and let $v_1, v_2 \in V$. If $v_1 \equiv_{\downarrow}^{k+1} v_2$, then $v_1 \equiv_{\downarrow}^{k} v_2$.*

Proof. It suffices to observe that "$\equiv_{\downarrow}^{k+1}$" is an equivalence relation satisfying Proposition 4.6 for the value of k in the statement of the Corollary, above. For the first two conditions in Proposition 4.6, this follows immediately from the corresponding conditions in Definition 4.3. For the third condition in Proposition 4.6, this also follows from the third condition in Definition 4.3 if one takes into account that, for arbitrary sets A and B, $\min(|A|, k+1) = \min(|B|, k+1)$ implies that $\min(|A|, k) = \min(|B|, k)$. □

4.2.2. Upward distinguishability

If we only look upward in the document, there is only one reasonable definition of node distinguishability, as each node has at most one parent. In contrast with the downward case, the recursion in the definition is on the *depth* of the first node.

Definition 4.8. Let $D = (V, Ed, r, \lambda)$ be a document, and let $v_1, v_2 \in V$. Then, v_1 and v_2 are *upward-equivalent*, denoted $v_1 \equiv_{\uparrow} v_2$, if

1. $\lambda(v_1) = \lambda(v_2)$;
2. v_1 is the root if and only if v_2 is the root;
3. if v_1 and v_2 are not the root, and u_1 and u_2 are the parents of v_1 and v_2, respectively, then $u_1 \equiv_{\uparrow} u_2$.

It is easily seen that two nodes are upward-equivalent if the paths from the root to these two nodes are isomorphic in the sense that they have the same length and corresponding nodes have the same label.

Example 4.9. In the example document of Figure 1 we have, e.g., that $v_6 \equiv_{\uparrow} v_7$, $v_8 \equiv_{\uparrow} v_9$, $v_{11} \equiv_{\uparrow} v_{12}$, but $v_8 \not\equiv_{\uparrow} v_{13}$.

4.2.3. Two-way distinguishability

If we look both upward and downward in a document, we can define a notion of equivalence by combining the definitions of upward- and k-downward-equivalence: two nodes are k-equivalent if they are upward-equivalent, and if corresponding nodes on the isomorphic paths from the root to these nodes are k-downward-equivalent. More formally, we have the following recursive definition, where the recursion is on the depth of the first node.

Definition 4.10. Let $D = (V, Ed, r, \lambda)$ be a document, let $v_1, v_2 \in V$, and let $k \geq 1$. Then, v_1 and v_2 are *k-equivalent*, denoted $v_1 \equiv_{\updownarrow}^{k} v_2$, if

1. $v_1 \equiv_{\downarrow}^{k} v_2$;
2. v_1 is the root if and only if v_2 is the root; and
3. if v_1 and v_2 are not the root, and u_1 and u_2 are the parents of v_1 and v_2, respectively, then $u_1 \equiv_{\updownarrow}^{k} u_2$.

Stated in a nonrecursive way, two nodes are k-equivalent if the paths from the root to these two nodes have equal length and corresponding nodes on these two paths are downward-k-equivalent.

Example 4.11. Consider again the example document in Figure 1. We have that, e.g, $v_5 \equiv_{\updownarrow}^{1} v_6 \equiv_{\updownarrow}^{1} v_7$, but no two of these nodes are k-equivalent for any value of $k \geq 2$. Also, $v_5 \not\equiv_{\updownarrow}^{k} v_8$ and $v_8 \not\equiv_{\updownarrow}^{k} v_{13}$, for any value of $k \geq 1$.

By a straightforward inductive argument, the following is immediate from Corollary 4.7.

Proposition 4.12. *Let $k \geq 1$. Let $D = (V, Ed, r, \lambda)$ be a document, and let $v_1, v_2 \in V$. If $v_1 \equiv_{\updownarrow}^{k+1} v_2$, then $v_1 \equiv_{\updownarrow}^{k} v_2$.*

4.3. Distinguishability of pairs of nodes at the syntactic level

We also define notions of distinguishability of *pairs* of nodes, by requiring that the pairs have subsumed or congruent signatures and that corresponding nodes on the (undirected) paths between begin and end points of both pairs are related under one of the notions defined in Subsection 4.2.

Definition 4.13. Let $D = (V, Ed, r, \lambda)$ be a document, let ϑ be one of the syntactic relationships between nodes defined in Subsection 4.2, and let v_1, w_1, v_2, and w_2 be nodes in V. Then, (v_1, w_1) *ϑ-subsumes* (v_2, w_2), denoted $(v_1, w_1) \gtrsim_\vartheta (v_2, w_2)$ (respectively, (v_1, w_1) and (v_2, w_2) are *ϑ-congruent*, denoted $(v_1, w_1) \cong_\vartheta (v_2, w_2)$) if

1. $(v_1, w_1) \gtrsim (v_2, w_2)$ (respectively, $(v_1, w_1) \cong (v_2, w_2)$); and
2. for each node y_1 on the path form v_1 to w_1, $y_1 \vartheta y_2$, where y_2 is the unique ancestor of v_2 or w_2 or both for which $(v_2, y_2) \in \mathrm{sig}(v_1, y_1)(D)$ (or, equivalently, $(y_2, w_2) \in \mathrm{sig}(y_1, w_1)(D)$).[6]

Example 4.14. Consider again the example document in Figure 1. We have that, e.g., $(v_2, v_5) \cong_{\equiv^k_\downarrow} (v_3, v_6)$ for $k = 1$ but not for any higher value of k; $(v_2, v_5) \cong_{\equiv^k_\downarrow} (v_{10}, v_{13})$ for any value of $k \geq 1$; $(v_2, v_5) \cong_{\equiv_\uparrow} (v_4, v_9)$; $(v_5, v_6) \cong_{\equiv^k_\updownarrow} (v_5, v_7)$ for any value of $k \geq 1$; and $(v_6, v_7) \gtrsim_{\equiv^1_\updownarrow} (v_2, v_5)$, but not the other way around.

The following observation is obvious from the definition.

Proposition 4.15. *Let $D = (V, Ed, r, \lambda)$ be a document, let $\varphi \in \{\gtrsim, \cong\}$, let ϑ be one of the syntactic relationships between nodes defined in Subsection 4.2, and let v_1, w_1, v_2, and w_2 be nodes of D such that $(v_1, w_1)\ \varphi_\vartheta\ (v_2, w_2)$. Let y_1 and y_2 be nodes on the path from v_1 to w_1, and let z_1 and z_2 be ancestors of v_2 or w_2 or both corresponding to y_1 and y_2, respectively. Then $(y_1, z_1)\ \varphi_\vartheta\ (y_2, z_2)$.*

The mutual position of the nodes in the statement of Proposition 4.15 is illustrated in Figure 3.

From Proposition 3.4, (1), the following is also obvious.

Proposition 4.16. *Let $D = (V, Ed, r, \lambda)$ be a document, let ϑ be one of the syntactic relationships between nodes defined in Subsection 4.2, and let v_1, v_2, w_1, and w_2 be nodes in V. If v_1 is an ancestor of w_1 or vice versa, $(v_1, w_1) \cong_\vartheta (v_2, w_2)$ if and only if $(v_1, w_1) \gtrsim_\vartheta (v_2, w_2)$.*

Finally, from Definitions 4.10 and 4.13, the following is immediate.

Proposition 4.17. *Let $D = (V, Ed, r, \lambda)$ be a document, let $v_1, v_2 \in V$, and let $k \geq 1$. Then, $v_1 \equiv^k_\updownarrow v_2$ if and only if $(r, v_1) \cong_{\equiv^k_\downarrow} (r, v_2)$.*

Table 3 summarizes all of the distinguishability notions presented in this section. The balance of the paper is devoted to identifying the languages which correspond in expressive power to each of these notions.

[6] In the sequel, we call y_1 and y_2 *corresponding* nodes.

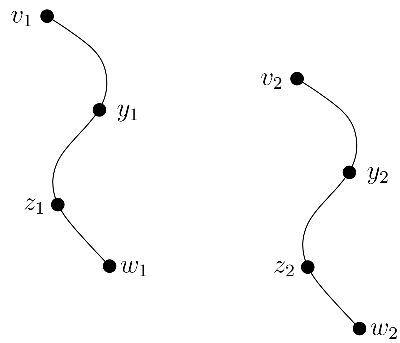

Figure 3: Mutual position of the nodes mentioned in the statement of Proposition 4.15.

Table 3: Distinguishability notions of Section 4.

distinguishability notion	*notation*	*defined in*
expression-related	$\geq_{\mathrm{exp}}$	Definition 4.1
expression-equivalent	$\equiv_{\mathrm{exp}}$	Definition 4.1
downward-k-equivalent	$\equiv_{\downarrow}^{k}$	Definition 4.3
upward-equivalent	$\equiv_{\uparrow}$	Definition 4.8
k-equivalent	$\equiv_{\updownarrow}^{k}$	Definition 4.10
ϑ-subsumes	$\gtrsim_{\vartheta}$	Definition 4.13
ϑ-congruent	$\cong_{\vartheta}$	Definition 4.13

5. Strictly downward languages

We call a language *downward* if, for any expression e in that language, and for any node v of the document D under consideration, all nodes in $e(D)(v)$ are descendants of v.

In this section, we consider languages with the stronger property that, for any expression e in the language, and for any node v of the document D under consideration, $e(D)(v) = e(D')(v)$, where D' is the subtree of D rooted at v. We shall call such languages *strictly downward*.

Downward languages that are *not* strictly downward will be called *weakly downward* and are the subject of Section 6.

Considering the nonbasic operations in Table 1, the language $\mathcal{X}(E)$ is strictly downward if and only if E does *not* contain upward navigation ("$\uparrow$"), second projection ("π_2"), and inverse ("$.^{-1}$"). It is the purpose of this section to investigate the expressive power of these languages at the document level, both for query expressiveness and navigational expressiveness, and, in some cases, derive actual characterizations for these.

5.1. Sufficient conditions for expression equivalence

If e is an expression in a downward language $\mathcal{X}(E)$, then it follows immediately from the definition that, given a node v of the document D under consideration, each node in $e(D)(v)$ is a descendant of v. Therefore, we only need to consider ancestor-descendant pairs of nodes, for which corresponding notions of subsumption and congruence coincide (Proposition 4.16).

The following property of $\equiv_{\downarrow}^{k}$-congruence, $k \geq 1$, for ancestor-descendant pairs of nodes will turn out to be very useful.

Lemma 5.1. *Let $D = (V, Ed, r, \lambda)$ be a document, let v_1, w_1, and v_2 be nodes of D such that w_1 is a descendant of v_1, and let $k \geq 1$. If $v_1 \equiv_{\downarrow}^{k} v_2$, then v_2 has a descendant w_2 in D such that $(v_1, w_1) \cong_{\equiv_{\downarrow}^{k}} (v_2, w_2)$.*

Proof. The proof is by induction of the length of the path from v_1 to w_1. If $w_1 = v_1$, then, obviously, Lemma 5.1 is satisfied for $w_2 := v_2$. If $w_1 \neq v_1$, then let y_1 be the child of v_1 on the path to w_1. By Definition 4.3, v_2 has a child y_2 such that $y_1 \equiv_{\downarrow}^{k} y_2$. By the induction hypothesis, y_2 has a descendant w_2 in D such that $(y_1, w_1) \cong_{\equiv_{\downarrow}^{k}} (y_2, w_2)$. From Definition 4.13, it is now straightforward that $(v_1, w_1) \cong_{\equiv_{\downarrow}^{k}} (v_2, w_2)$. □

We now link $\equiv_{\downarrow}^{k}$-congruence of ancestor-descendant pairs of nodes with expressibility in strictly downward languages.

Proposition 5.2. *Let $k \geq 1$, and let E be the set of all nonbasic operations in Table 1, except for upward navigation ("$\uparrow$"), second projection ("π_2"), inverse ("$.^{-1}$"), and selection on at least m children satisfying some condition ("$ch_{\geq m}(.)$") for $m > k$. Let e be an expression in $\mathcal{X}(E)$. Let $D = (V, Ed, r, \lambda)$ be a document, let v_1, w_1, v_2, and w_2 be nodes of D such that w_1 is a descendant of v_1 and w_2 is a descendant of v_2. Assume furthermore that $(v_1, w_1) \cong_{\equiv_{\downarrow}^{k}} (v_2, w_2)$. Then, $(v_1, w_1) \in e(D)$ if and only if $(v_2, w_2) \in e(D)$.*

Proof. By symmetry, it suffices to show that $(v_1, w_1) \in e(D)$ implies $(v_2, w_2) \in e(D)$. We prove this by structural induction. For the atomic operators $\emptyset$, ε, $\hat{\ell}$ ($\ell \in \mathcal{L}$), and $\downarrow$, it is straightforward that Proposition 5.2 holds. We have now settled the base case and turn to the induction step.

1. $e := e_1/e_2$, with e_1 and e_2 satisfying Proposition 5.2. Assume that $(v_1, w_1) \in e(D)$. Then there exists $y_1 \in V$ such that $(v_1, y_1) \in e_1(D)$ and $(y_1, w_1) \in e_2(D)$. By the strictly downward nature of $\mathcal{X}(E)$, y_1 is on the path from v_1 to w_1. Let y_2 be the node on the path from v_2 to w_2 corresponding to y_1. By Proposition 4.15, $(v_1, y_1) \equiv_{\downarrow}^{k} (v_2, y_2)$ and $(y_1, w_1) \equiv_{\downarrow}^{k} (y_2, w_2)$. By the induction hypothesis, $(v_2, y_2) \in e_1(D)$ and $(y_2, w_2) \in e_2(D)$. Hence, $(v_2, w_2) \in e(D)$.
2. $e := \pi_1(f)$, with f satisfying Proposition 5.2. Assume that $(v_1, w_1) \in e(D)$. Then, necessarily $v_1 = w_1$, and, consequently, $v_2 = w_2$. From $(v_1, v_1) \in \pi_1(f)(D)$, it follows that there exists $z_1 \in V$ such that $(v_1, z_1) \in f(D)$. Since $v_1 \equiv_{\downarrow}^{k} v_2$, it also follows, by Lemma 5.1, that there exists

a descendant z_2 of w_2 such that $(v_1, z_1) \cong_{\equiv_\downarrow^k} (v_2, z_2)$. By the induction hypothesis, $(v_2, z_2) \in f(D)$. Hence, $(v_2, v_2) \in e(D)$.

3. $e := \mathrm{ch}_{\geq m}(f)$, with $m \leq k$ and f satisfying Proposition 5.2. Assume that $(v_1, w_1) \in \mathrm{ch}_{\geq m}(f)(D)$. Hence, $v_1 = w_1$, which in turn implies $v_2 = w_2$. Let $\downarrow/\pi_1(f)(D)(v_1) = Y_1$ and let $\downarrow/\pi_1(f)(D)(v_2) = Y_2$. By assumption, $|Y_1| \geq m$. Now, let y be a child of v_1 in Y_1 or a child of v_2 in Y_2, and let z be a child of v_1 not in Y_1 or a child of v_2 not in Y_2. By assumption, there exists a node y' such that $(y, y') \in f(D)$. Now, suppose that $y \equiv_\downarrow^k z$. Then, by Proposition 5.1, there exists a node z' such that $(y, y') \cong_{\equiv_\downarrow^k} (z, z')$. But then, by the induction hypothesis, $(z, z') \in f(D)$, contrary to our assumptions. We may therefore conclude that $y \not\equiv_\downarrow^k z$. Since furthermore $v_1 \equiv_\downarrow^k v_2$, it follows that, for all $y_1 \in Y_1$, there exists $y_2 \in Y_2$ such that $y_1 \equiv_\downarrow^k y_2$, and vice versa. Hence, for some $n \geq 1$, we can write $Y_1 = Y_{11} \cup \ldots \cup Y_{1n}$ and $Y_2 = Y_{21} \cup \ldots \cup Y_{2n}$ such that
   (a) $Y_{11}, \ldots, Y_{1n}$ are maximal sets of mutually downward-k-equivalent children of v_1, and are hence pairwise disjoint;
   (b) $Y_{21}, \ldots, Y_{2n}$ are maximal sets of mutually downward-k-equivalent children of v_2, and are hence pairwise disjoint; and
   (c) for all $i = 1, \ldots, n$, each node of Y_{1i} is downward-k-equivalent to each node of Y_{2i}.

   If, for some i, $|Y_{1i}| \geq k$, it follows from $v_1 \equiv_\downarrow^k v_2$ that $|Y_{2i}| \geq k$, and, hence, that $|Y_2| \geq k \geq m$. If, on the other hand, for all $i = 1, \ldots, n$, $|Y_{1i}| < k$, it follows from $v_1 \equiv_\downarrow^k v_2$ that $|Y_{1i}| = |Y_{2i}|$, and, hence, that $|Y_1| = |Y_2|$. Since $|Y_1| \geq m$, it follows that, also in this case, $|Y_2| \geq m$. We may thus conclude that, in all cases, $|Y_2| \geq m$, and, hence, that $(v_2, v_2) \in \mathrm{ch}_{\geq m}(f)(D) = e(D)$.
4. $e := e_1 \cup e_2$, with e_1 and e_2 satisfying Proposition 5.2. Assume that $(v_1, w_1) \in e(D)$. Then, $(v_1, w_1) \in e_1(D)$ or $(v_1, w_1) \in e_2(D)$. Without loss of generality, assume the former. Then, by the induction hypothesis, $(v_2, w_2) \in e_1(D)$. Hence, $(v_2, w_2) \in e(D)$.
5. $e := e_1 \cap e_2$, with e_1 and e_2 satisfying Proposition 5.2. Assume that $(v_1, w_1) \in e(D)$. Then, $(v_1, w_1) \in e_1(D)$ and $(v_1, w_1) \in e_2(D)$. It follows by the induction hypothesis that $(v_2, w_2) \in e_1(D)$ and $(v_2, w_2) \in e_2(D)$. Hence, $(v_2, w_2) \in e(D)$.
6. $e := e_1 - e_2$, with e_1 and e_2 satisfying Proposition 5.2. Assume that $(v_1, w_1) \in e(D)$. Then $(v_1, w_1) \in e_1(D)$ and $(v_1, w_1) \notin e_2(D)$. By the induction hypothesis, $(v_2, w_2) \in e_1(D)$ and $(v_2, w_2) \notin e_2(D)$. (Indeed, if $(v_2, w_2) \in e_2(D)$, then, again by the induction hypothesis, $(v_1, w_1) \in e_2(D)$, a contradiction.) Hence, $(v_2, w_2) \in e(D)$.

$\square$

Corollary 5.3. *Let $k \geq 1$, and let E be the set of all nonbasic operations in Table 1, except for upward navigation ("$\uparrow$"), second projection ("π_2"), inverse ("$.^{-1}$"), and selection on at least m children ("$\mathrm{ch}_{\geq m}(.)$") for $m > k$. Let e be an*

expression in $\mathcal{X}(E)$. Let $D = (V, Ed, r, \lambda)$ be a document, let v_1 and v_2 be nodes of D such that $v_1 \equiv_{\downarrow}^{k} v_2$ and let w_1 be a descendant of v_1. If $(v_1, w_1) \in e(D)$, then there exists a descendant w_2 of v_2 such that $(v_2, w_2) \in e(D)$.

Proof. By Lemma 5.1, there exists a descendant w_2 of v_2 such that $(v_1, w_1) \cong_{\equiv_{\downarrow}^{k}} (v_2, w_2)$. By Proposition 5.2, it now follows that $(v_2, w_2) \in e(D)$. $\square$

Corollary 5.4. *Let $k \geq 1$, and let E be a set of nonbasic operations in Table 1 not containing upward navigation ("$\uparrow$"), second projection ("π_2"), inverse ("$.^{-1}$"), or selection on at least m children satisfying some condition ("$ch_{\geq m}(.)$") for $m > k$. Consider the language $\mathcal{X}(E)$ or $\mathcal{C}(E)$. Let $D = (V, Ed, r, \lambda)$ be a document, and let v_1 and v_2 be nodes of D. If $v_1 \equiv_{\downarrow}^{k} v_2$, then $v_1 \equiv_{\text{exp}} v_2$.*

Proof. Let e be an expression in the language under consideration such that $e(D)(v_1) \neq \emptyset$. Hence, there exists a descendant w_1 of v_1 such that $(v_1, w_1) \in e(D)$. Notice that e is also an expression in the language considered in Corollary 5.3. Hence, there exists a descendant w_2 of v_2 such that $(v_2, w_2) \in e(D)$, so $e(D)(v_2) \neq \emptyset$. By symmetry, the converse also holds. We may thus conclude that $v_1 \equiv_{\text{exp}} v_2$. $\square$

We may thus conclude that downward-k-equivalence is a sufficient condition for expression-equivalence under a strictly downward language provided $\text{ch}_{\geq m}$ cannot be expressed for $m > k$.

Even more, Corollary 5.4 does no longer hold if this restriction is removed, as shown by the following counterexample.

Example 5.5. Consider again the example document in Figure 1. We established in Example 4.4 that $v_2 \equiv_{\downarrow}^{1} v_3$, but $v_2 \not\equiv_{\downarrow}^{2} v_3$. In the language $\mathcal{X}(\text{ch}_{\geq 2})$, clearly $v_2 \not\equiv_{\text{exp}} v_3$, as $\text{ch}_{\geq 2}(\varepsilon)(D)(v_2) = \emptyset$, while $\text{ch}_{\geq 2}(\varepsilon)(D)(v_3) \neq \emptyset$.

5.2. Necessary conditions for expression equivalence

We now explore requirements on the set of nonbasic operations expressible in the language under which downward-k-equivalence ($k \geq 1$) is a necessary condition for expression-equivalence. As we have endeavored to make as few assumptions as possible, Proposition 5.6 also holds for a class of languages that are *not* (strictly) downward.

Proposition 5.6. *Let $k \geq 1$, and let E be a set of nonbasic operations containing set difference ("$-$"). Consider the language $\mathcal{X}(E)$ or $\mathcal{C}(E)$. Assume that, in this language, first projection ("π_1") can be expressed, as well as selection on at least m children satisfying some condition ("$ch_{\geq m}(.)$"), for all $m = 1, \ldots, k$. Let $D = (V, Ed, r, \lambda)$ be a document, and let v_1 and v_2 be nodes of D. If $v_1 \equiv_{\text{exp}} v_2$, then $v_1 \equiv_{\downarrow}^{k} v_2$.*

Proof. Since expression-equivalence in the context of $\mathcal{X}(E)$ implies expression-equivalence in the context of $\mathcal{C}(E)$, we may assume without loss of generality that the language under consideration is $\mathcal{C}(E)$. To prove Proposition 5.6, it

suffices to show that expression-equivalence ("$\equiv_{\text{exp}}$") satisfies the conditions of Proposition 4.6.

1. If $v_1 \equiv_{\text{exp}} v_2$, then $\lambda(v_1) = \lambda(v_2)$, for, otherwise, $\widehat{\lambda(v_1)}(D)(v_1) \neq \emptyset$, while $\widehat{\lambda(v_1)}(D)(v_2) = \emptyset$, a contradiction.
2. If $v_1 \equiv_{\text{exp}} v_2$ and v_1 is not a leaf, then v_2 is not a leaf either, for, otherwise, $\text{ch}_{\geq 1}(\varepsilon)(D)(v_1) \neq \emptyset$, while $ch_1(\varepsilon)(D)(v_2) = \emptyset$, a contradiction. Let w_1 be a child of v_1, and let $w_2^1, \ldots, w_2^n$ be all children of w_2. Suppose for the sake of contradiction that, for all $i = 1, \ldots, n$, $w_1 \not\equiv_{\text{exp}} w_2^i$. Then, by Proposition 4.2, there exists an expression e_i in $\mathcal{C}(E)$ such that $e_i(D)(w_1) \neq \emptyset$ and $e_i(D)(w_2^i) = \emptyset$, for all $i = 1, \ldots, n$. Now, let $e := \pi_1(e_1) \cap \ldots \cap \pi_1(e_n)$, which can be expressed in $\mathcal{C}(E)$.[7] Then, $\text{ch}_{\geq 1}(e)(D)(v_1) \neq \emptyset$ while $\text{ch}_{\geq 1}(e)(D)(v_2) = \emptyset$, contradicting $v_1 \equiv_{\text{exp}} v_2$. Hence, there does exist a child w_2 of v_2 such that $w_1 \equiv_{\text{exp}} w_2$. Of course, the same also goes with the roles of v_1 and v_2 reversed.
3. Finally, let v_1 and v_2 be non-leaf nodes such that $v_1 \equiv_{\text{exp}} v_2$, and let w_1 and w_2 be children of v_1 and v_2, respectively, such that $w_1 \equiv_{\text{exp}} w_2$. For $i = 1, 2$, let $\tilde{w}_i$ be the set of all siblings of w_i (including w_i itself) that are expression-equivalent to w_i. As in the previous item, we can construct an expression e in $\mathcal{C}(E)$ such that $e(D)(w_1) \neq \emptyset$ (and hence $e(D)(w) \neq \emptyset$ for each node w in $\tilde{w}_1$ or $\tilde{w}_2$) and $e(D)(w) = \emptyset$ for each sibling of w_1 not in $\tilde{w}_1$ *and* for each sibling of w_2 not in $\tilde{w}_2$. For the sake of contradiction, assume that $\min(|\tilde{w}_1|, k) \neq \min(|\tilde{w}_2|, k)$. Without loss of generality, assume that $\min(|\tilde{w}_1|, k) < \min(|\tilde{w}_2|, k)$. Hence, $\min(|\tilde{w}_1|, k) = |\tilde{w}_1|$. Let $m := \min(|\tilde{w}_2|, k)$. Then, $\text{ch}_{\geq m}(e)(D)(v_1) = \emptyset$, while $\text{ch}_{\geq m}(e)(D)(v_2) \neq \emptyset$, contradicting $v_1 \equiv_{\text{exp}} v_2$. We may thus conclude that $\min(|\tilde{w}_1|, k) = \min(|\tilde{w}_2|, k)$.

$\square$

Notice that the languages satisfying the statement of Proposition 5.6 need not contain any navigation operations ("$\downarrow$" or "$\uparrow$"). Of course, in the context of this Section, we are interested in languages in which downward navigation ("$\downarrow$") is possible. Specializing Proposition 5.6 to this case, we may thus conclude that downward-k-equivalence is a necessary condition for expression-equivalence under a strictly downward language containing first projection ("π_1") and set difference ("$-$"), provided selection on at least m children satisfying some condition ("$\text{ch}_{\geq m}$") for all $m = 1, \ldots, k$ can be expressed.

5.3. Characterization of expression equivalence

The languages containing downward navigation ("$\downarrow$") and satisfying both Corollary 5.4 of Subsection 5.1 and Proposition 5.6 of Subsection 5.2 are $\mathcal{X}(\downarrow, \pi_1, \text{ch}_{\geq 1}(.), \ldots, \text{ch}_{\geq k}(.), -)$ and $\mathcal{C}(\downarrow, \pi_1, \text{ch}_{\geq 1}(.), \ldots, \text{ch}_{\geq k}(.), -)$. We call

[7] Let f_1 and f_2 be expressions in $\mathcal{C}(E)$ such that $f_1(D) \subseteq \varepsilon(D)$ and $f_2(D) \subseteq \varepsilon(D)$. Then, $f_1 \cap f_2$ can be expressed in $\mathcal{C}(E)$ as $\pi_1(\varepsilon - \pi_1(\varepsilon - f_1) \cup \pi_1(\varepsilon - f_2))$.

these languages the *strictly downward XPath algebra with counting up to* k and the *strictly downward core XPath algebra with counting up to* k, respectively. Combining the aforementioned results, we get the following.

Theorem 5.7. *Let* $k \geq 1$*, and consider the strictly downward (core) XPath algebra with counting up to* k*. Let* $D = (V, Ed, r, \lambda)$ *be a document, and let* v_1 *and* v_2 *be nodes of* D*. Then* $v_1 \equiv_{\mathrm{exp}} v_2$*, if and only if* $v_1 \equiv_{\downarrow}^{k} v_2$*.*

A special case arises when $k = 1$, since selection on at least one child satisfying some condition ("$\mathrm{ch}_{\geq 1}(.)$") can be expressed in terms of the other operations required by Theorem 5.7, by Proposition 2.4. The languages we then obtain, $\mathcal{X}(\downarrow, \pi_1, -)$ and $\mathcal{C}(\downarrow, \pi_1, -)$, are called the *strictly downward XPath algebra* and the *strictly downward core XPath algebra*, respectively. We have the following.

Corollary 5.8. *Consider the strictly downward (core) XPath algebra. Let* $D = (V, Ed, r, \lambda)$ *be a document, and let* v_1 *and* v_2 *be nodes of* D*. Then* $v_1 \equiv_{\mathrm{exp}} v_2$*, if and only if* $v_1 \equiv_{\downarrow}^{1} v_2$*.*

5.4. Characterization of navigational expressiveness

We shall now investigate the expressiveness of strictly downward languages at the document level. In other words, we shall address the question whether, given a document, we can characterize when a set of pairs of nodes of that document is the result of some query in the language under consideration applied to that document. Such type of results are often referred to as BP-characterizations, after Bancilhon [22] and Paredaens [23] who first proved such results for Codd's relational calculus and algebra, respectively (cf. [24]).

We start by proving a converse to Proposition 5.2.

Proposition 5.9. *Let* $k \geq 1$*, and let* E *be a set of nonbasic operations containing downward navigation ("*$\downarrow$*") and set difference ("*$-$*"). Consider the language* $\mathcal{X}(E)$ *or* $\mathcal{C}(E)$*. Assume that, in this language, first projection ("*π_1*") can be expressed, as well as selection on at least* m *children satisfying some condition ("*$\mathrm{ch}_{\geq m}(.)$*"), for all* $m = 1, \ldots, k$*. Let* $D = (V, Ed, r, \lambda)$ *be a document, and let* v_1*,* w_1*,* v_2*, and* w_2 *be nodes of* D *such that* w_1 *is a descendant of* v_1 *and* w_2 *is a descendant of* v_2*. Assume furthermore that, for each expression* e *in the language,* $(v_1, w_1) \in e(D)$ *if and only if* $(v_2, w_2) \in e(D)$*. Then* $(v_1, w_1) \cong_{\equiv_{\downarrow}^{k}} (v_2, w_2)$*.*

Proof. First notice that, by assumption, $(v_2, w_2) \in \mathrm{sig}(v_1, w_1)(D)$, and vice versa. Hence, $(v_1, w_1) \cong (v_2, w_2)$. Let y_1 be a node on the path from v_1 to w_1, and let y_2 be the corresponding node on the path from v_2 to w_2. By construction, $(v_1, y_1) \cong (v_1, y_2)$ and $(y_1, w_1) \cong (y_2, w_2)$. Now, let f be any expression in the language such that $f(D)(y_1) \neq \emptyset$. Then, $(y_1, y_1) \in \pi_1(f)(D)$. Let $e := \mathrm{sig}(v_1, y_1)/\pi_1(f)/\mathrm{sig}(y_1, w_1)$. By construction, $(v_1, w_1) \in e(D)$. Hence, by assumption, $(v_2, w_2) \in e(D)$, which implies $(y_2, y_2) \in \pi_2(f)(D)$ or $f(D)(y_2) \neq \emptyset$. The same holds vice versa, and we may thus conclude that $y_1 \equiv_{\mathrm{exp}} y_2$, and, hence, by Proposition 5.6, $y_1 \equiv_{\downarrow}^{k} y_2$. We may thus conclude that $(v_1, w_1) \equiv_{\downarrow}^{k} (v_2, w_2)$. $\square$

Combining Propositions 5.2 and 5.9, we obtain the following.

Corollary 5.10. *Let $k \geq 1$, and consider the strictly downward (core) XPath algebra with counting up to k. Let $D = (V, Ed, r, \lambda)$ be a document, and let v_1, w_1, v_2, and w_2 be nodes of D such that w_1 is a descendant of v_1 and w_2 is a descendant of v_2. Then, the property that, for each expression e in the language under consideration, $(v_1, w_1) \in e(D)$ if and only if $(v_2, w_2) \in e(D)$ is equivalent to the property $(v_1, w_1) \cong_{\equiv^k_\downarrow} (v_2, w_2)$.*

In order to state our first BP-result, we need the following two lemmas.

Lemma 5.11. *Let $k \geq 1$. Let $D = (V, Ed, r, \lambda)$ be a document, and let v_1 be a node of D. There exists an expression e_{v_1} in the strictly downward core XPath algebra with counting up to k such that, for each node v_2 of D, $e_{v_1}(D)(v_2) \neq \emptyset$ if and only if $v_1 \equiv^k_\downarrow v_2$.*

Proof. Let w be any node of D such that $v_1 \not\equiv^k_\downarrow w$. By Theorem 5.7, $v_1 \not\equiv_{\text{exp}} w$. By Proposition 4.2, there exists an expression $f_{v_1,w}$ in the strictly downward core XPath algebra with counting up to k such that $f_{v_1,w}(D)(v_1) \neq \emptyset$ and $f_{v_1,w}(D)(w) = \emptyset$. Now consider the expression

$$e_{v_1} := \pi_1\left(\bigcap_{w \in V \ \& \ v_1 \not\equiv^k_\downarrow w} \pi_1(f_{v_1,w})\right),$$

which is also in the strictly downward core XPath algebra with counting up to k. By construction, $e_{v_1}(D)(v_1) \neq \emptyset$. Now consider a node v_2 of D. If $v_1 \equiv^k_\downarrow v_2$, then, by Theorem 5.7, $v_1 \equiv_{\text{exp}} v_2$. Hence, by definition, $e_{v_1}(D)(v_2) \neq \emptyset$. If, on the other hand, $v_1 \not\equiv^k_\downarrow v_2$, then, by construction, $e_{v_1}(D)(v_2) = \emptyset$. □

Lemma 5.12. *Let $k \geq 1$. Let $D = (V, Ed, r, \lambda)$ be a document, and let v_1 and w_1 be nodes of D such that w_1 is a descendant of v_1. There exists an expression e_{v_1,w_1} in the strictly downward core XPath algebra with counting up to k such that, for all nodes v_2 and w_2 of D with w_2 a descendant of v_2, $(v_2, w_2) \in e_{v_1,w_1}(D)$ if and only if $(v_1, w_1) \cong_{\equiv^k_\downarrow} (v_2, w_2)$.*

Proof. From Lemma 5.11, we know that, for node y_1 of D, there exists an expression e_{y_1} in the strictly downward core XPath algebra with counting up to k such that, for each node y_2 of D, $e_{y_1}(D)(y_2) \neq \emptyset$ if and only if $y_1 \equiv^k_\downarrow y_2$. Now, let v_1 and w_1 be nodes of D such that w_1 is a descendant of v_1, and let $v_1 = y_{11}, \ldots, y_{1n} = w_1$ be the path from v_1 to w_1 in D. Define

$$e_{v_1,w_1} := \pi_1(e_{y_{11}})/\downarrow/\pi_1(e_{y_{12}})/\ldots\downarrow/\pi_1(e_{y_{1n}}),$$

which is also in the strictly downward core XPath algebra with counting up to k. By construction, $(v_1, w_1) \in e_{v_1,w_1}(D)$. Let v_2 and w_2 be nodes of D such that w_2 is a descendant of v_2. If $(v_1, w_1) \cong_{\equiv^k_\downarrow} (v_2, w_2)$, then, by Corollary 5.10, $(v_2, w_2) \in e_{v_1,w_1}(D)$. Conversely, if $(v_2, w_2) \in e_{v_1,w_1}(D)$, then, by construction,

$(v_1, w_1) \cong (v_2, w_2)$. Thus, let $v_2 = y_{21}, \dots, y_{2n} = w_2$ be the path from v_2 to w_2 in D. Again by construction, it follows that, for $j = 1, \dots, n$, $e_{y_{1j}}(D)(y_{2j}) \neq \emptyset$, or, equivalently, that $y_{1j} \equiv_{\downarrow}^{k} y_{2j}$. Hence, $(v_1, w_1) \equiv_{\downarrow}^{k} (v_2, w_2)$. ☐

We are now ready to state the actual result.

Theorem 5.13. *Let $k \geq 1$. Let $D = (V, Ed, r, \lambda)$ be a document, and let $R \subseteq V \times V$. Then, there exists an expression e in the strictly downward (core) XPath algebra with counting up to k such that $e(D) = R$ if and only if,*

1. *for all $v, w \in V$, $(v, w) \in R$ implies w is a descendant of v; and,*
2. *for all $v_1, w_1, v_2, w_2 \in V$ with w_1 a descendant of v_1, w_2 a descendant of v_2, and $(v_1, w_1) \cong_{\equiv_{\downarrow}^{k}} (v_2, w_2)$, $(v_1, w_1) \in R$ implies $(v_2, w_2) \in R$.*

Proof. To see the "only if," it suffices to notice that the first condition follows from the downward character of the language, and the second from Corollary 5.10. The remainder of the proof concerns the "if." From Lemma 5.12, we know that, for all nodes v_1 and w_1 of D such that w_1 is a descendant of v_1, there exists an expression e_{v_1,w_1} in $\mathcal{C}(E)$ such that, for all nodes v_2 and w_2 of D, $(v_2, w_2) \in e_{v_1,w_1}(D)$ if and only if $(v_1, w_1) \cong_{\equiv_{\downarrow}^{k}} (v_2, w_2)$. Now consider the expression

$$e := \bigcup_{(v_1,w_1) \in R} e_{v_1,w_1}.$$

This expression, which is well defined because $(v_1, w_1) \in R$ by assumption implies that w_1 is a descendant of v_1, is also in $\mathcal{C}(E)$ (and hence also in $\mathcal{X}(E)$). It remains to show that $e(D) = R$. Clearly, $R \subseteq e(D)$. We prove the reverse inclusion. Thereto, let v_2 and w_2 be nodes such that $(v_2, w_2) \in e(D)$. By construction, there exist nodes v_1 and w_1 in D such that w_1 is a descendant of v_1 and $(v_2, w_2) \in e_{v_1,w_1}(D)$. Hence, $(v_1, w_1) \cong_{\equiv_{\downarrow}^{k}} (v_2, w_2)$. But then, by assumption, also $(v_2, w_2) \in R$. So, $e(D) \subseteq R$. ☐

As before, we can specialize Theorem 5.13 to the strictly downward (core) XPath algebra.

Corollary 5.14. *Let $D = (V, Ed, r, \lambda)$ be a document, and let $R \subseteq V \times V$. There exists an expression e in the strictly downward (core) XPath algebra such that $e(D) = R$ if and only if,*

1. *for all $v, w \in V$, $(v, w) \in R$ implies w is a descendant of v;*
2. *for all $v_1, w_1, v_2, w_2 \in V$ with w_1 a descendant of v_1, w_2 a descendant of v_2, and $(v_1, w_1) \equiv_{\downarrow}^{1} (v_2, w_2)$, $(v_1, w_1) \in R$ implies $(v_2, w_2) \in R$.*

We can also recast Theorem 5.13 in terms of node-level navigation.

Theorem 5.15. *Let $k \geq 1$. Let $D = (V, Ed, r, \lambda)$ be a document, let v be a node of D, and let $W \subseteq V$. Then there exists an expression e in the strictly downward (core) XPath algebra with counting up to k such that $e(D)(v) = W$ if and only if all nodes of W are descendants of v, and, for all $w_1, w_2 \in W$ with $(v, w_1) \cong_{\equiv_{\downarrow}^{k}} (v, w_2)$, $w_1 \in W$ implies $w_2 \in W$.*

Proof. Only if. Let e be an expression in the language under consideration such that $e(D)(v) = W$. Let $w_1, w_2 \in V$ be descendants of v with $(v, w_1) \equiv_{\downarrow}^{k} (v, w_2)$, and assume that $w_1 \in W = e(D)(v)$. Hence, $(v, w_1) \in e(D)$. By Corollary 5.10, $(v, w_2) \in e(D)$. Hence, $w_2 \in e(D)(v) = W$.

If. Let $W \subseteq V$ satisfy the property that all nodes of W are descendants of v, and, for all $w_1, w_2 \in V$ with $w_1 \equiv_{\downarrow}^{k} w_2$, $w_1 \in W$ implies $w_2 \in W$. Let $R := \{(v', w_2) \mid \text{ there exists } w_1 \in W \text{ such that } (v, w_1) \cong_{\equiv_{\downarrow}^{k}} (v', w_2)\}$. Clearly, R satisfies the properties of Theorem 5.13. Hence, there exists an expression e in the language under consideration such that $R = e(D)$. Clearly, $W \subseteq e(D)(v)$. We prove the reverse inclusion. Therefore, let $w_2 \in e(D)(v)$, i.e., $(v, w_2) \in R$. Then there exists $w_1 \in R$ such that $(v, w_1) \cong_{\equiv_{\downarrow}^{k}} (v, w_2)$. By the property that W satisfies, $w_2 \in W$. Hence, $e(D)(v) \subseteq W$, and, therefore, $e(D)(v) = W$. $\square$

Again, we can specialize Theorem 5.15 to the strictly downward (core) XPath algebra.

Corollary 5.16. *Let $D = (V, Ed, r, \lambda)$ be a document, let v be a node of D, and let $W \subseteq V$. Then there exists an expression e in the strictly downward (core) XPath algebra such that $e(D)(v) = W$ if and only if all nodes of W are descendants of v, and, for all nodes w_1 and w_2 of D with $(v, w_1) \cong_{\equiv_{\downarrow}^{1}} (v, w_2)$, $w_1 \in W$ implies $w_2 \in W$.*

A special case of Theorem 5.16 is when we are only interested in navigation from the *root*.

Theorem 5.17. *Let $k \geq 1$. Let $D = (V, Ed, r, \lambda)$ be a document, and let $W \subseteq V$. Then there exists an expression e in the strictly downward (core) XPath algebra with counting up to k such that $e(D)(r) = W$ if and only if, for all nodes w_1 and w_2 of D with $w_1 \equiv_{\updownarrow}^{k} w_2$, $w_1 \in W$ implies $w_2 \in W$.*

Proof. From Theorem 5.15, it immediately follows that there exists an expression e in the language under consideration such that $e(D)(r) = W$ if and only if, for $w_1, w_2 \in V$ with $(r, w_1) \cong_{\equiv_{\downarrow}^{k}} (r, w_2)$, $w_1 \in W$ implies $w_2 \in W$. By Proposition 4.17, $(r, w_1) \cong_{\equiv_{\downarrow}^{k}} (r, w_2)$ is equivalent to $w_1 \equiv_{\updownarrow}^{k} w_2$. $\square$

The specialization of Theorem 5.17 to the case of the strictly downward (core) XPath algebra is as follows.

Corollary 5.18. *Let $D = (V, Ed, r, \lambda)$ be a document, and let $W \subseteq V$. Then there exists an expression e in the strictly downward (core) XPath algebra such that $e(D)(r) = W$ if and only if, for all nodes w_1 and w_2 of D with $w_1 \equiv_{\updownarrow}^{1} w_2$, $w_1 \in W$ implies $w_2 \in W$.*

To conclude this section, we observe that none of the characterization results above distinguish between the language $\mathcal{X}(E)$ and the corresponding core language $\mathcal{C}(E)$. This is not surprising, as, for *all* downward languages, they have the same expressive power, not only at the navigational level for a given document, but also at the level of queries, i.e., for each expression e in $\mathcal{X}(E)$,

there exists an equivalent expression e' in the corresponding core language $\mathcal{C}(E)$, meaning that, *for each* document D, $e(D) = e'(D)$. Thereto, we prove a slightly stronger result.

Theorem 5.19. *Let E be a set of nonbasic operations containing downward navigation ("$\downarrow$") and first projection ("π_1"), and not containing upward navigation ("$\uparrow$"), and inverse ("$.^{-1}$"). Let e be an expression in the language under consideration. With the exception of intersection ("$\cap$") and set difference ("$-$") operations used as operands in boolean combinations of subexpressions of the language within a first projection or conditional operation, all intersection and set difference operations can be eliminated, to the extent that these operations occur in the language under consideration.*

Proof. The proof goes by structural induction. Therefore, consider the expression $e_1 \cap e_2$, respectively, $e_1 - e_2$ (to the extent these operations occur in the language under consideration), where e_1 and e_2 are expressions not containing eliminable intersection and set difference operations. For $i = 1, 2$, we may write

$$e_i = c_{i0}/\downarrow/c_{i1}/\downarrow/\ldots/\downarrow/c_{in_i-1}/\downarrow/c_{in_i},$$

where, for $j = 0, \ldots, n_i$, c_{ij} is an expression in $\mathcal{C}(E)$ with the property that, for each document D, $c_{ij}(D) \subseteq \varepsilon(D)$. From here on, we consider both cases separately.

1. *Intersection.* Clearly, if $n_1 \neq n_2$, then, for each document D, $e_1 \cap e_2(D) = \emptyset = \emptyset(D)$. In the other case, let $n := n_1 = n_2$. For $j = 0, \ldots, n$, let $c_j := \pi_1(c_{1j} \cap c_{2j})$, which is an expression of $\mathcal{C}(E)$, equivalent to $c_{1j} \cap c_{2j}$. Let
$$e' := c_0/\downarrow/c_1/\downarrow/\ldots/\downarrow/c_{n-1}/\downarrow/c_n.$$
A straightforward set-theoretical argument reveals that, for each document D, $e'(D) = e_1 \cap e_2(D)$.
2. *Difference.* Clearly, if $n_1 \neq n_2$, then, for each document D, $e_1 - e_2(D) = e_1(D)$. In the other case, let $n := n_1 = n_2$. For $j = 0, \ldots, n$, let e'_j be e_1 in which c_{1j} is replaced by $\pi_1(c_{1j} - c_{2j})$, which is an expression of $\mathcal{C}(E)$, equivalent to $c_{1j} - c_{2j}$. Let
$$e' = e'_0 \cup e'_1 \cup \ldots \cup e'_{n-1} \cup e'_n.$$
which is also in $\mathcal{C}(E)$. A straightforward set-theoretical argument reveals that, for each document D, $e'(D) = e_1 - e_2(D)$.

$\square$

Corollary 5.20. *Let E be a set of nonbasic operations containing downward navigation ("$\downarrow$") and first projection ("π_1"), and not containing upward navigation ("$\uparrow$"), and inverse ("$.^{-1}$"). Then, for each expression e in $\mathcal{X}(E)$, there exists an expression e' in $\mathcal{C}(E)$ such that, for each document D, $e(D) = e'(D)$.*

By Theorem 5.19, we may even disallow set difference or intersection operations (to the extent they occur in the language under consideration) except those used as operands of boolean combinations of subexpressions inside a projection operation without loosing expressive power.

5.5. Strictly downward languages not containing set difference

So far, the characterizations of strictly downward languages involved only languages containing the set difference operator. One could, therefore, wonder if it is possible to provide similar characterizations for languages not containing set difference. However, the absence of set difference and the logical negation that is inherently embedded in it has as a side effect that it is no longer always possible to exploit equivalences or derive them.

5.5.1. Weaker notions of downward and two-way distinguishability

Therefore, one would like to consider an asymmetric version of downward k-equivalence, say "downward k-relatedness," which, for the appropriate language could correspond to expression relatedness. For $k = 1$, such an approach could lead to the following definitions.

Definition 5.21. Let $D = (V, Ed, r, \lambda)$ be a document, and let $v_1, v_2 \in V$. Then,

1. v_1 and v_2 are *downward-related*, denoted $v_1 \geq_\downarrow v_2$, if
   (a) $\lambda(v_1) = \lambda(v_2)$; and
   (b) for each child w_1 of v_1, there exists a child w_2 of v_2 such that $w_1 \geq_\downarrow w_2$.
2. v_1 and v_2 are *weakly downward-equivalent*, denoted $v_1 \approxeq_\downarrow v_2$, if $v_1 \geq_\downarrow v_2$ and $v_2 \geq_\downarrow v_1$.

Obviously, downward 1-equivalence implies weak downward equivalence. The converse, however, is *not* true, as illustrated by the following, simple example.

Example 5.22. Consider the document in Figure 4. Labels have been omitted, because they are not relevant in this discussion. (We assume all nodes have the same label.) Obviously, $x_1 \equiv^1_\downarrow x_2$, hence $x_1 \approxeq_\downarrow x_2$. In particular, $x_1 \geq_\downarrow x_2$ and $x_2 \geq_\downarrow x_1$. Also, $y_1 \geq_\downarrow x_2$, as the second condition to be verified is voidly satisfied in this case. We may thus conclude that $v_1 \approxeq_\downarrow v_2$. However, $v_1 \not\equiv^1_\downarrow v_2$, as there is no child of v_2 that is downward 1-equivalent to y_1.

Notice that, in Example 5.22, there is even no child of v_2 that is *weakly* downward equivalent to y_1! Therefore, we shall not even attempt to generalize Definition 5.21 to the case where $k > 1$, as there is no straightforward way to adapt the third condition of Definition 4.3.

We conclude this digression on alternatives for downward 1-equivalence by providing analogue alternatives for 1-equivalence.

Definition 5.23. Let $D = (V, Ed, r, \lambda)$ be a document, and let $v_1, v_2 \in V$. Then,

1. v_1 and v_2 are *related*, denoted $v_1 \geq_\updownarrow v_2$, if
   (a) $v_1 \geq_\downarrow v_2$;
   (b) v_1 is the root if and only if v_2 is the root; and

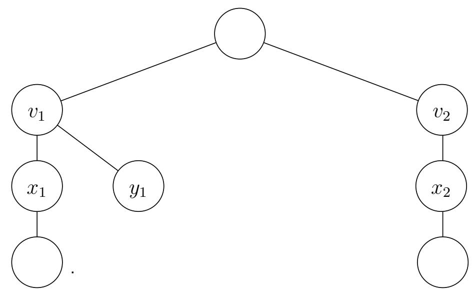

Figure 4: Document of Example 5.22.

Table 4: Distinguishability notions of Section 5.5.1.

distinguishability notion	*notation*	*defined in*
downward-related	$\geq_{\downarrow}$	Definition 5.21
weakly-downward-equivalent	$\approxeq_{\downarrow}$	Definition 5.21
related	$\geq_{\updownarrow}$	Definition 5.23
weakly-equivalent	$\approxeq_{\updownarrow}$	Definition 5.23

   (c) if v_1 and v_2 are not the root, and u_1 and u_2 are the parents of v_1 and v_2, respectively, then $u_1 \geq_{\updownarrow} u_2$.

2. v_1 and v_2 are *weakly equivalent*, denoted $v_1 \approxeq_{\updownarrow} v_2$, if $v_1 \geq_{\updownarrow} v_2$ and $v_2 \geq_{\updownarrow} v_1$.

Example 5.24. Consider again the document in Figure 4. Observe that $v_1 \approxeq_{\updownarrow} v_2$. Furthermore, $y_1 \geq_{\updownarrow} x_2$, but not the other way around.

Table 4 summarizes all of the distinguishability notions presented in this section.

The following analogue of Proposition 4.17 is straightforward.

Proposition 5.25. *Let $D = (V, Ed, r, \lambda)$ be a document, and let $v_1, v_2 \in V$. Then,*

1. $v_1 \geq_{\updownarrow} v_2$ *if and only if* $(r, v_1) \cong_{\geq_{\downarrow}} (r, v_2)$*; and*
2. $v_1 \approxeq_{\updownarrow} v_2$ *if and only if* $(r, v_1) \cong_{\approxeq_{\downarrow}} (r, v_2)$*.*

5.5.2. Towards characterizing expression equivalence and navigational expressiveness

The approach we shall take here is reviewing the results in Sections 5.1–5.3 and examine to which extent these results in the case where $k = 1$ allow replacing downward 1-equivalence by weak downward equivalence.

We start by observing that the analogue of Lemma 5.1 does not hold. Indeed, in the example document of Example 5.22, shown in Figure 4, $v_1 \approxeq_{\downarrow} v_2$. Also,

there is no child of v_2 that is weakly downward equivalent to x_1. Hence, there is no node node z for which $(v_1, x_1) \cong_{\approxeq^1_\downarrow} (v_2, z)$. On the other hand, we can restrict Lemma 5.1 to downward relatedness:

Lemma 5.26. *Let $D = (V, Ed, r, \lambda)$ be a document, let v_1, w_1, and v_2 be nodes of D such that w_1 is a descendant of v_1. If $v_1 \geq_\downarrow v_2$, then v_2 has a descendant w_2 in D such that $(v_1, w_1) \cong_{\geq_\downarrow} (v_2, w_2)$.*

Proposition 5.2 relies on Lemma 5.1 to prove the inductive step for the first projection ("π_1"). It therefore comes as no surprise that we cannot replace downward 1-equivalence by weak downward equivalence, there. Indeed, consider the expression $e := \pi_1(\downarrow/(\varepsilon - \pi_1(\downarrow)))$. In the example document of Example 5.22, shown in Figure 4, $v_1 \approxeq_\downarrow v_2$, and, hence, $(v_1, v_1) \cong_{\approxeq_\downarrow} (v_2, v_2)$. Moreover, $(v_1, v_1) \in e(D)$. However, $(v_2, v_2) \notin e(D)$. However, we can "save" Proposition 5.2 by replacing downward 1-equivalence by downward *relatedness*, provided we omit set difference ("$-$") from the set of operations of the language. Indeed, we can then recover the proof, using Lemma 5.26 instead of Lemma 5.1. (Notice that, for the induction step for set difference in the original proof, we must exploit equivalence in both directions to deal with the negation inherent to the difference operation.) In summary, we have the following.

Lemma 5.27. *Let E be the set of all nonbasic operations in Table 1, except for upward navigation ("$\uparrow$"), second projection ("π_2"), inverse ("$.^{-1}$"), selection on at least k children satisfying some condition ("$ch_{\geq k}(.)$") for $k > 1$, and set difference ("$-$"). Let e be an expression in $\mathcal{X}(E)$. Let $D = (V, Ed, r, \lambda)$ be a document, let v_1, w_1, v_2, and w_2 be nodes of D such that w_1 is a descendant of v_1 and w_2 is a descendant of v_2. Assume furthermore that $(v_1, w_1) \cong_{\geq_\downarrow} (v_2, w_2)$. Then, $(v_1, w_1) \in e(D)$ implies $(v_2, w_2) \in e(D)$.*

Two applications of Lemma 5.27 immediately yield the following.

Proposition 5.28. *Let E be the set of all nonbasic operations in Table 1, except for upward navigation ("$\uparrow$"), second projection ("π_2"), inverse ("$.^{-1}$"), selection on at least k children satisfying some condition ("$ch_{\geq k}(.)$") for $k > 1$, and set difference ("$-$"). Let e be an expression in $\mathcal{X}(E)$. Let $D = (V, Ed, r, \lambda)$ be a document, let v_1, w_1, v_2, and w_2 be nodes of D such that w_1 is a descendant of v_1 and w_2 is a descendant of v_2. Assume furthermore that $(v_1, w_1) \cong_{\approxeq_\downarrow} (v_2, w_2)$. Then, $(v_1, w_1) \in e(D)$ if and only $(v_2, w_2) \in e(D)$.*

So, Proposition 5.28 is weaker than Proposition 5.2 in the sense that we had to exclude set difference, but stronger in the sense that, in return, we were able to replace the precondition by a weaker one.

The analogues of Corollaries 5.3 and 5.4 are now as follows.

Corollary 5.29. *Let E be the set of all nonbasic operations in Table 1, except for upward navigation ("$\uparrow$"), second projection ("π_2"), inverse ("$.^{-1}$"), selection on at least k children satisfying some condition ("$ch_{\geq k}(.)$") for $k > 1$, and set difference ("$-$"). Let e be an expression in $\mathcal{X}(E)$. Let $D = (V, Ed, r, \lambda)$ be*

a document, let v_1 and v_2 be nodes of D such that $v_1 \geq_{\downarrow} v_2$ and let w_1 be a descendant of v_1. If $(v_1, w_1) \in e(D)$, then there exists a descendant w_2 of v_2 such that $(v_2, w_2) \in e(D)$.

In other words, downward relatedness implies expression relatedness.

Corollary 5.30. *Let E be a set of nonbasic operations not containing upward navigation ("$\uparrow$"), second projection ("π_2"), inverse ("$.^{-1}$"), selection on at least k children satisfying some condition ("$ch_{\geq k}(.)$") for $k > 1$, and set difference ("$-$"). Consider the language $\mathcal{X}(E)$ or $\mathcal{C}(E)$. Let $D = (V, Ed, r, \lambda)$ be a document, and let v_1 and v_2 be nodes of D. If $v_1 \approxeq_{\downarrow} v_2$, then $v_1 \equiv_{\exp} v_2$.*

Proof. The condition $v_1 \approxeq_{\downarrow} v_2$ implies $v_1 \geq_{\downarrow} v_2$ and $v_2 \geq_{\downarrow} v_1$. By Corollary 5.29, these conditions in turn imply $v_1 \geq_{\exp} v_2$ and $v_2 \geq_{\exp} v_1$, which together are equivalent to $v_1 \equiv_{\exp} v_2$. $\square$

We now look to necessary conditions for expression equivalence for strictly downward languages not containing set difference. Provided intersection ("$\cap$") is available, the expressibility of set difference is used only once in the proof of Proposition 5.6, namely where Proposition 4.2 is invoked. We do not need this Proposition, however, in the following variation of Proposition 5.6:

Lemma 5.31. *Let E be a set of nonbasic operations containing first projection ("π_1"), and intersection ("$\cap$"). Consider the language $\mathcal{X}(E)$ or $\mathcal{C}(E)$. Let $D = (V, Ed, r, \lambda)$ be a document, and let v_1 and v_2 be nodes of D. If $v_1 \geq_{\exp} v_2$, then $v_1 \geq_{\downarrow} v_2$.*

Two applications of Lemma 5.31 immediately yield the following.

Proposition 5.32. *Let E be a set of nonbasic operations containing first projection ("π_1"), and intersection ("$\cap$"). Consider the language $\mathcal{X}(E)$ or $\mathcal{C}(E)$. Let $D = (V, Ed, r, \lambda)$ be a document, and let v_1 and v_2 be nodes of D. If $v_1 \equiv_{\exp} v_2$, then $v_1 \approxeq_{\downarrow} v_2$.*

So, Proposition 5.32 is weaker than Proposition 5.6 in the sense that the conclusion is replaced by a weaker one, but stronger in the sense that, in return, we no longer have to rely on the presence of difference.

The languages containing downward navigation ("$\downarrow$") and satisfying both Corollary 5.30 and Proposition 5.32 are $\mathcal{X}(\downarrow, \pi_1, \cap)$ and $\mathcal{C}(\downarrow, \pi_1, \cap)$, which, moreover, are equivalent, by Corollary 5.20. In addition, we can eliminate intersection operations except those used as operands of boolean combinations of subexpressions inside a projection operation without loosing expressive power. We call these languages the *strictly downward positive XPath algebra* and the *strictly downward core positive XPath algebra*, respectively. Combining the aforementioned results, we get the following.

Theorem 5.33. *Consider the strictly downward (core) positive XPath algebra. Let $D = (V, Ed, r, \lambda)$ be a document, and let v_1 and v_2 be nodes of D. Then $v_1 \equiv_{\exp} v_2$ if and only if $v_1 \approxeq_{\downarrow} v_2$.*

We finally turn to the characterization of navigational expressiveness. Proposition 5.9 and its proof, and hence also Corollary 5.10, carry over to the current setting.

Theorem 5.34. *Consider the strictly downward (core) positive XPath algebra. Let $D = (V, Ed, r, \lambda)$ be a document, and let v_1, w_1, v_2, and w_2 be nodes of D such that w_1 is a descendant of v_1 and w_2 is a descendant of v_2. Then, the property that, for each expression e in the language under consideration, $(v_1, w_1) \in e(D)$ if and only if $(v_2, w_2) \in e(D)$ is equivalent to the property $(v_1, w_1) \cong_{\approxeq_\downarrow} (v_2, w_2)$.*

To derive a BP-result for the strictly downward (core) positive XPath algebra, we observe that Lemmas 5.11 and 5.12 and Theorem 5.13 carry over to the current context, provided we replace downward 1-equivalence by downward relatedness.

Lemma 5.35. *Let $D = (V, Ed, r, \lambda)$ be a document.*

1. *Let v_1 be a node of D. There exists an expression e_{v_1} in the strictly downward (core) positive XPath algebra such that, for each node v_2 of D, $e_{v_1}(D)(v_2) \neq \emptyset$ if and only if $v_1 \geq_\downarrow v_2$.*
2. *Let v_1 and w_1 be a nodes of D such that w_1 is a descendant of v_1. There exists an expression $e_{(v_1,w_1)}$ in the strictly downward (core) positive XPath algebra such that, for all nodes v_2 and w_2 of D with w_2 a descendant of v_2, $(v_2, w_2) \in e_{(v_1,w_1)}(D)$ if and only if $(v_1, w_1) \cong_{\geq_\downarrow} (v_2, w_2)$.*

In the proof of the first claim, the role of Theorem 5.7 is taken over by Corollary 5.29.

Theorem 5.36. *Let $D = (V, Ed, r, \lambda)$ be a document, and let $R \subseteq V \times V$. Then, there exists an expression e in the strictly downward (core) positive XPath algebra such that $e(D) = R$ if and only if,*

1. *for all $v, w \in V$, $(v, w) \in R$ implies w is a descendant of v; and,*
2. *for all $v_1, w_1, v_2, w_2 \in V$ with w_1 a descendant of v_1, w_2 a descendant of v_2, and $(v_1, w_1) \cong_{\geq_\downarrow} (v_2, w_2)$, $(v_1, w_1) \in R$ implies $(v_2, w_2) \in R$.*

The major difference between Theorems 5.13 and 5.36 is that, in the former, R is a partition of maximal sets of $\equiv_\downarrow^k$-congruent nodes, while, in the latter, R is merely closed under $\geq_\downarrow$-congruence.

We can also recast Theorem 5.36 in terms of node-level navigation, in much the same way as Theorem 5.13.

Theorem 5.37. *Let $D = (V, Ed, r, \lambda)$ be a document, let v be a node of D, and let $W \subseteq V$. Then there exists an expression e in the strictly downward (core) positive XPath algebra such that $e(D)(v) = W$ if and only if all nodes of W are descendants of v, and, for all nodes w_1 and w_2 of D with $(v, w_1) \cong_{\geq_\downarrow} (v, w_2)$, $w_1 \in W$ implies $w_2 \in W$.*

Corollary 5.38. *Let $D = (V, Ed, r, \lambda)$ be a document, and let $W \subseteq V$. Then there exists an expression e in the strictly downward (core) positive XPath algebra such that $e(D)(r) = W$ if and only if, for all nodes w_1 and w_2 of D with $w_1 \geq_{\updownarrow} w_2$, $w_1 \in W$ implies $w_2 \in W$.*

For the last result of this section, we relied on Proposition 5.25, (1).

6. Weakly downward languages

We now turn to *weakly downward* languages: for any node v of the document D under consideration, all nodes in $e(D)(v)$ are descendants of v, but there are possibly nodes v for which $e(D)(v) \neq e(D')(v)$, with D' the subtree of D rooted at v.

6.1. Sufficient conditions for expression-equivalence

The key notion in Sections 6.1–6.3 is $\equiv^k_{\updownarrow}$-congruence, $k \geq 1$, restricted to ancestor-descendant pairs. We first explore some properties of this notion.

Lemma 6.1. *Let $D = (V, Ed, r, \lambda)$ be a document, let v_1, w_1, v_2, and w_2 be nodes of D such that w_1 is a descendant of v_1 and w_2 is a descendant of v_2, and let $k \geq 1$. Then, $(v_1, w_1) \cong_{\equiv^k_{\updownarrow}} (v_2, w_2)$ if and only if $(v_1, w_1) \cong (v_2, w_2)$ and $w_1 \equiv^k_{\updownarrow} w_2$.*

Proof. As the "only if" is obvious, we focus on the "if." By Proposition 4.17, $w_1 \equiv^k_{\updownarrow} w_2$ implies that $(r, w_1) \cong_{\equiv^k_{\downarrow}} (r, w_2)$. Let y_1 be a node on the path from v_1 to w_1, and let y_2 be the corresponding node on the path from v_2 to w_2. By Proposition 4.15, we also have that $(r, y_1) \cong_{\equiv^k_{\downarrow}} (r, y_2)$. By another application of Proposition 4.17, we finally deduce that $y_1 \equiv^k_{\updownarrow} y_2$. □

Lemma 6.2. *Let $D = (V, Ed, r, \lambda)$ be a document, let v_1 and w_1 be nodes of D such that w_1 is a descendant of v_1, and let $k \geq 1$. Then,*

1. *each node v_2 of D for which $v_1 \equiv^k_{\updownarrow} v_2$ has a descendant w_2 in D such that $(v_1, w_1) \cong_{\equiv^k_{\updownarrow}} (v_2, w_2)$; and*
2. *each node w_2 of D for which $w_1 \equiv^k_{\updownarrow} w_2$ has an an ancestor v_2 in D such that $(v_1, w_1) \cong_{\equiv^k_{\updownarrow}} (v_2, w_2)$.*

Proof. To see (1), we know by Lemma 5.1 that v_2 has a descendant w_2 such that $(v_1, w_1) \cong_{\equiv^k_{\downarrow}} (v_2, w_2)$. By Proposition 4.17, we also have that $(r, v_1) \cong_{\equiv^k_{\downarrow}} (r, v_2)$. It now readily follows that $(r, w_1) \cong_{\equiv^k_{\downarrow}} (r, w_2)$, or, again by Proposition 4.17, $w_1 \equiv^k_{\updownarrow} w_2$. It now follows from Lemma 6.1 that $(v_1, w_1) \cong_{\equiv^k_{\updownarrow}} (v_2, w_2)$.

Claim (2) can be shown by induction on the length of the path from v_1 to w_1. If $v_1 = w_1$, then obviously, we must choose $v_2 := w_2$. If $v_1 \neq w_1$, we have in particular that $w_1 \neq r$, and, hence, by $w_1 \equiv^k_{\updownarrow} w_2$, that $w_2 \neq r$. Let y_1 be the parent of w_1 and y_2 be the parent of w_2. By definition, $y_1 \equiv^k_{\updownarrow} y_2$, and, by the

induction hypothesis there is a node v_2 in D such that $(v_1, y_1) \cong_{\equiv^k_\updownarrow} (v_2, y_2)$. It now readily follows that $(v_1, w_1) \cong_{\equiv^k_\updownarrow} (v_2, w_2)$. □

We now link $\equiv^k_\updownarrow$-congruence of ancestor-descendant pairs of nodes with expressibility in weakly downward languages.

Proposition 6.3. *Let $k \geq 1$, and let E be the set of all nonbasic operations in Table 1, except for upward navigation ("$\uparrow$"), inverse ("$.^{-1}$"), and selection on at least m children satisfying some condition ("$ch_{\geq m}(.)$") for $m > k$. Let e be an expression in $\mathcal{X}(E)$. Let $D = (V, Ed, r, \lambda)$ be a document, and let v_1, w_1, v_2, and w_2 be nodes of D such that w_1 is a descendant of v_1 and w_2 is a descendant of v_2. Assume furthermore that $(v_1, w_1) \cong_{\equiv^k_\updownarrow} (v_2, w_2)$. Then, $(v_1, w_1) \in e(D)$ if and only if $(v_2, w_2) \in e(D)$.*

Proof. The proof goes along the same lines of the proof of Proposition 5.2. Actually, since $\equiv^k_\updownarrow$-congruence implies $\equiv^k_\downarrow$-congruence, almost all of the proof by structural induction can be used here verbatim, except, of course, for the inductive step for the second projection ("π_2"), which we consider next. Thus, let $e := \pi_2(f)$, with f satisfying Proposition 6.3. If $(v_1, w_1) \in \pi_2(f)$, then, of course, $v_1 = w_1$ as a consequence of which $v_2 = w_2$. Also, there exists $y_1 \in V$ such that $(y_1, v_1) \in f(D)$. By Lemma 6.2, (2), there exists $y_2 \in V$ such that $(y_1, v_1) \cong_{\equiv^k_\updownarrow} (y_2, v_2)$. By the induction hypothesis, $(y_2, v_2) \in f(D)$. Hence, $(v_2, v_2) \in \pi_2(f)(D)$. □

By combining Proposition 6.3 with Lemma 6.2, we can establish the following.

Corollary 6.4. *Let $k \geq 1$, and let E be the set of all nonbasic operations in Table 1, except for upward navigation ("$\uparrow$"), inverse ("$.^{-1}$"), and selection on at least m children ("$ch_{\geq m}(.)$") for $m > k$. Let e be an expression in $\mathcal{X}(E)$. Let $D = (V, Ed, r, \lambda)$ be a document, let v_1 and w_1 be nodes of D such that w_1 is a descendant of v_1 and $(v_1, w_1) \in e(D)$. Then,*

1. *each node v_2 of D for which $v_1 \equiv^k_\updownarrow v_2$ has a descendant w_2 in D such that $(v_2, w_2) \in e(D)$; and*
2. *each node w_2 of D for which $w_1 \equiv^k_\updownarrow w_2$ has an an ancestor v_2 in D such that $(v_2, w_2) \in e(D)$.*

Finally, we infer the following from Corollary 5.4, (1):

Corollary 6.5. *Let $k \geq 1$, and let E be a set of nonbasic operations not containing upward navigation ("$\uparrow$"), inverse ("$.^{-1}$"), and selection on at least m children satisfying some condition ("$ch_{\geq m}(.)$") for $m > k$. Consider the language $\mathcal{X}(E)$ or $\mathcal{C}(E)$. Let $D = (V, Ed, r, \lambda)$ be a document, and let v_1 and v_2 be nodes of D. If $v_1 \equiv^k_\updownarrow v_2$, then $v_1 \equiv_{\mathrm{exp}} v_2$.*

6.2. Necessary conditions for expression equivalence

We now explore requirements on the set of nonbasic operations expressible in the language under which downward-k-equivalence ($k \geq 1$) is a necessary condition for expression-equivalence. As we have endeavored to make as few assumptions as possible, Proposition 6.6 also holds for a class of languages that are *not* downward.

Proposition 6.6. *Let $k \geq 1$, and let E be a set of nonbasic operations containing at least one navigation operation ("$\downarrow$" or "$\uparrow$") and set difference ("$-$"). Consider the language $\mathcal{X}(E)$ or $\mathcal{C}(E)$, and assume that, in this language, first and second projection ("π_1" and "π_2") can be expressed, as well as selection on at least m children satisfying some condition ("$\mathrm{ch}_{\geq m}(.)$"), for all $m = 1, \ldots, k$. Let $D = (V, Ed, r, \lambda)$ be a document, and let v_1 and v_2 be nodes of D. If $v_1 \equiv_{\mathrm{exp}} v_2$, then $v_1 \equiv^k_{\updownarrow} v_2$.*

Proof. Without loss of generality, we may assume that the language under consideration is $\mathcal{C}(E)$. In Proposition 5.6, we have already established that $v_1 \equiv_{\mathrm{exp}} v_2$ implies $v_1 \equiv^k_{\downarrow} v_2$. By induction on the length of the path from r to v_1, we establish that, furthermore, $v_1 \equiv^k_{\updownarrow} v_2$. For the basis of the induction, consider the case that $v_1 = r$. Let d be the length of a longest path from r to a leaf of D (i.e., the height of the tree). We distinguish two cases:

1. $\downarrow \in E$. Then, $\downarrow^d(D)(v_1) \neq \emptyset$. Hence, $\downarrow^d(D)(v_2) \neq \emptyset$, which implies $v_2 = r$.
2. $\uparrow \in E$ Then, $\pi_2(\uparrow^d)(D)(v_1) \neq \emptyset$. Hence, $\pi_2(\uparrow^d)(D)(v_2) \neq \emptyset$, which implies $v_2 = r$.

In both cases, it follows that $v_1 \equiv^k_{\updownarrow} v_2$. For the induction step, consider the case that $v_1 \neq r$. Again, we distinguish two cases:

1. $\downarrow \in E$. Then, $\pi_2(\downarrow)(D)(v_1) \neq \emptyset$, and, hence, $\pi_2(\downarrow)(D)(v_2) \neq \emptyset$. So, $v_2 \neq r$.
2. $\uparrow \in E$. Then, $\uparrow(D)(v_1) \neq \emptyset$, and, hence, $\uparrow(D)(v_2) \neq \emptyset$. So, $v_2 \neq r$.

Now, let u_1 be the parent of v_1 and u_2 be the parent of v_2. We show that $u_1 \equiv_{\mathrm{exp}} u_2$. Thereto, let e be an expression in the language under consideration for which $e(D)(u_1) \neq \emptyset$. Again, we distinguish two cases:

1. $\downarrow \in E$. Then, $\pi_2(e/\downarrow)(v_1) \neq \emptyset$. Since $v_1 \equiv_{\mathrm{exp}} v_2$, $\pi_2(e/\downarrow)(v_2) \neq \emptyset$. It follows that $e(D)(u_2) \neq \emptyset$.
2. $\uparrow \in E$. Then, $\uparrow/e(v_1) \neq \emptyset$. Since $v_1 \equiv_{\mathrm{exp}} v_2$, $\uparrow/e(v_2) \neq \emptyset$. It follows that $e(D)(u_2) \neq \emptyset$.

By the induction hypothesis, we may now conclude that, in both cases, $u_1 \equiv^k_{\updownarrow} u_2$. Hence, also $v_1 \equiv^k_{\updownarrow} v_2$. $\square$

We see that Proposition 6.6 is as well applicable to weakly downward languages as to weakly upward languages (see Section 7.2). We shall see in Section 7.2 that this is no coincidence. For now, we suffice with concluding that k-equivalence is a necessary condition for expression-equivalence under a weakly downward language containing downward navigation ("$\downarrow$"), both projections ("π_1" and "π_2"), and set difference ("$-$"), provided selection on at least m children satisfying some condition ("$\mathrm{ch}_{\geq m}$") for all $m = 1, \ldots, k$ can be expressed.

6.3. Characterization of expression equivalence

The weakly downward languages containing downward navigation ("$\downarrow$") and satisfying both Corollary 6.5 of Subsection 6.1 and Proposition 5.6 of Subsection 5.2 are

$$\mathcal{X}(\downarrow, \pi_1, \pi_2, \mathrm{ch}_{\geq 1}(.), \ldots, \mathrm{ch}_{\geq k}(.), -) \text{ and } \mathcal{C}(\downarrow, \pi_1, \pi_2, \mathrm{ch}_{\geq 1}(.), \ldots, \mathrm{ch}_{\geq k}(.), -),$$

which, moreover, are equivalent, by Corollary 5.20. In addition, we can eliminate set difference or intersection operations except those used as operands of boolean combinations of subexpressions inside a projection operation without loosing expressive power. We call these languages the *weakly downward XPath algebra with counting up to k* and the *weakly downward core XPath algebra with counting up to k*, respectively. Combining the aforementioned results, we get the following.

Theorem 6.7. *Let $k \geq 1$, and consider the weakly downward (core) XPath algebra with counting up to k. Let $D = (V, Ed, r, \lambda)$ be a document, and let v_1 and v_2 be nodes of D. Then $v_1 \equiv_{\exp} v_2$, if and only if $v_1 \equiv_{\updownarrow}^{k} v_2$.*

A special case arises when $k = 1$, since selection on at least one child satisfying some condition ("$\mathrm{ch}_{\geq 1}(.)$") can be expressed in terms of the other operations required by Theorem 6.7, by Proposition 2.4. The languages we then obtain are called the *weakly downward XPath algebra* and the *weakly downward core XPath algebra*, respectively. We have the following.

Corollary 6.8. *Consider the weakly downward (core) XPath algebra. Let $D = (V, Ed, r, \lambda)$ be a document, and let v_1 and v_2 be nodes of D. Then $v_1 \equiv_{\exp} v_2$, if and only if $v_1 \equiv_{\updownarrow}^{1} v_2$.*

6.4. Characterization of navigational expressiveness

We start by proving a converse to Proposition 6.3.

Proposition 6.9. *Let $k \geq 1$, and let E be a set of nonbasic operations containing downward navigation ("$\downarrow$") and set difference ("$-$"). Consider the language $\mathcal{X}(E)$ or $\mathcal{C}(E)$. Assume that, in this language, first and second projection ("π_1" and "π_2") can be expressed, as well as selection on at least m children satisfying some condition ("$\mathrm{ch}_{\geq m}(.)$"), for all $m = 1, \ldots, k$. Let $D = (V, Ed, r, \lambda)$ be a document, and let v_1, w_1, v_2, and w_2 be nodes of D such that w_1 is a descendant of v_1 and w_2 is a descendant of v_2. Assume furthermore that, for each expression e in the language, $(v_1, w_1) \in e(D)$ if and only if $(v_2, w_2) \in e(D)$. Then $(v_1, w_1) \cong_{\equiv_{\updownarrow}^{k}} (v_2, w_2)$.*

Proof. From Proposition 5.9, we already know that $(v_1, w_1) \cong_{\equiv_{\downarrow}^{k}} (v_2, w_2)$. In particular, $(v_1, w_1) \cong (v_2, w_2)$. By Lemma 6.1, it suffices to prove that $v_1 \equiv_{\updownarrow}^{k} w_2$, or, by Proposition 4.17, that $(r, w_1) \cong_{\equiv_{\downarrow}^{k}} (r, w_2)$. In view of what we already know, we only need to show that $(r, v_1) \cong_{\equiv_{\downarrow}^{k}} (r, v_2)$. Since $(v_1, w_1) \in \pi_2(\mathrm{sig}(r, v_1))/\mathrm{sig}(v_1, w_1)$, it follows that also $(v_2, w_2) \in \pi_2(\mathrm{sig}(r, v_1))/\mathrm{sig}(v_1, w_1)$,

for which we readily deduce that $(r, v_1) \cong (r, v_2)$. Let u_1 be a node on the path from r to v_1, and let u_2 be the corresponding node on the path from r to v_2. Then, $(r, u_1) \cong (r, u_2)$ and $(u_1, v_1) \cong (u_2, u_2)$. Now, let f be any expression in the language such that $f(D)(u_1) \neq \emptyset$. Then, $(u_1, u_1) \in \pi_1(f)(D)$. Let $e := \pi_2(\pi_1(f)/\mathrm{sig}(u_1, v_1))/\mathrm{sig}(v_1, w_1)$. By construction, $(v_1, w_1) \in e(D)$. Hence, by assumption, $(v_2, w_2) \in e(D)$, which implies $(u_2, u_2) \in \pi_1(f)(D)$ or $f(D)(u_2) \neq \emptyset$. The same holds vice versa, and we may thus conclude that $u_1 \equiv_{\mathrm{exp}} u_2$, and, hence, by Proposition 5.6, $u_1 \equiv^k_{\downarrow} u_2$. We may thus conclude that $(r, v_1) \equiv^k_{\downarrow} (r, v_2)$. □

Combining Propositions 6.3 and 6.9, we obtain the following.

Corollary 6.10. *Let $k \geq 1$, and consider the weakly downward (core) XPath algebra with counting up to k. Let $D = (V, Ed, r, \lambda)$ be a document, and let v_1, w_1, v_2, and w_2 be nodes of D such that w_1 is a descendant of v_1 and w_2 is a descendant of v_2. Then, the property that, for each expression e in the language under consideration, $(v_1, w_1) \in e(D)$ if and only if $(v_2, w_2) \in e(D)$ is equivalent to the property $(v_1, w_1) \cong_{\equiv^k_{\updownarrow}} (v_2, w_2)$.*

From here on, the derivation of a BP-result for the weakly downward (core) XPath algebra with counting up to k follows the development in Section 5.4 very closely, which is why we only state the final results.

Theorem 6.11. *Let $k \geq 1$. Let $D = (V, Ed, r, \lambda)$ be a document, and let $R \subseteq V \times V$. Then, there exists an expression e in the weakly downward (core) XPath algebra with counting up to k such that $e(D) = R$ if and only if,*

1. *for all $v, w \in V$, $(v, w) \in R$ implies w is a descendant of v; and,*
2. *for all $v_1, w_1, v_2, w_2 \in V$ with w_1 a descendant of v_1, w_2 a descendant of v_2, and $(v_1, w_1) \cong_{\equiv^k_{\updownarrow}} (v_2, w_2)$, $(v_1, w_1) \in R$ implies $(v_2, w_2) \in R$.*

The specialization to the weakly downward (core) XPath algebra is as follows.

Corollary 6.12. *Let $D = (V, Ed, r, \lambda)$ be a document, and let $R \subseteq V \times V$. There exists an expression e in the weakly downward (core) XPath algebra such that $e(D) = R$ if and only if,*

1. *for all $v, w \in V$, $(v, w) \in R$ implies w is a descendant of v; and,*
2. *for all $v_1, w_1, v_2, w_2 \in V$ with w_1 a descendant of v_1, w_2 a descendant of v_2, and $(v_1, w_1) \cong_{\equiv^1_{\updownarrow}} (v_2, w_2)$, $(v_1, w_1) \in R$ implies $(v_2, w_2) \in R$.*

We recast Theorem 6.11 and Corollary 6.12 in terms of node-level navigation.

Theorem 6.13. *Let $k \geq 1$. Let $D = (V, Ed, r, \lambda)$ be a document, let v be a node of D, and let $W \subseteq V$. Then there exists an expression e in the weakly downward (core) XPath algebra with counting up to k such that $e(D)(v) = W$ if and only if all nodes of W are descendants of v, and, for all $w_1, w_2 \in W$ with $(v, w_1) \cong_{\equiv^k_{\updownarrow}} (v, w_2)$, $w_1 \in W$ implies $w_2 \in W$.*

Corollary 6.14. *Let $D = (V, Ed, r, \lambda)$ be a document, let v be a node of D, and let $W \subseteq V$. Then there exists an expression e in the weakly downward (core) XPath algebra such that $e(D)(v) = W$ if and only if all nodes of W are descendants of v, and, for all $w_1, w_2 \in W$ with $(v, w_1) \cong_{\equiv^1_\updownarrow} (v, w_2)$, $w_1 \in W$ implies $w_2 \in W$.*

For $v = r$, the condition $(v, w_1) \cong_{\equiv^k_\updownarrow} (v, w_2)$ reduces to $w_1 \equiv^k_\updownarrow w_2$, by Proposition 4.17 and Lemma 6.1. Comparing Theorem 6.13 and Corollary 6.14 with, respectively, Theorem 5.17 and Corollary 5.18 then immediately yields the following.

Theorem 6.15. *Let $D = (V, Ed, r, \lambda)$.*

1. *for each expression e in the weakly downward (core) XPath algebra with counting up to k, $k \geq 1$, there exists an expression e' in the strictly downward (core) XPath algebra with counting up to k such that $e(D)(r) = e'(D)(r)$; in particular,*
2. *for each expression e in the weakly downward (core) XPath algebra, there exists an expression e' in the strictly downward (core) XPath algebra such that $e(D)(r) = e'(D)(r)$.*

Hence, the corresponding weakly downward and strictly downward languages are navigationally equivalent if navigation always starts from the root.

6.5. Weakly downward languages not containing set difference

To find characterizations for weakly downward languages not containing set difference, we can proceed in two ways:

1. we proceed as in Section 5.5.2 for strictly downward languages without set difference, i.e., reviewing the results in Sections 6.1–6.3 and examine to which extent these results in the case where $k = 1$ allow replacing 1-equivalence by relatedness (Definition 5.23); or
2. we start from the results in Section 5.5.2 on strictly downward languages without set difference and "bootstrap" them to results on weakly downward languages without set difference in the same way as the results on strictly downward languages with set difference in Sections 5.1–5.3 were bootstrapped to results on weakly down ward languages with set difference in Sections 6.1–6.3.

Of course, both approaches lead to the same results. As the necessary intermediate lemmas and all the proofs can readily be deduced in one of the two ways described above, we limit ourselves to giving the main results. Only one technical subtlety deserves mentioning here: despite the absence of difference, both the property that a node is the root and the property that a node is not the root can be expressed, the latter using second projection. For more details, we refer to the proof of Proposition 6.6.

Concretely, the language for which we provide characterizations in this Section, are $\mathcal{X}(\downarrow, \pi_1, \pi_2, \cap)$ and $\mathcal{C}(\downarrow, \pi_1, \pi_2, \cap)$, which , moreover, are equivalent, by

Corollary 5.20. We call these languages the *weakly downward positive XPath algebra* and the *weakly downward core positive XPath algebra*, respectively. In addition, we can eliminate intersection *altogether*. This follows from an earlier result by some of the present authors [31]. Although this result was stated in the context of languages that allow both downward and upward navigation, a careful examination of the elimination algorithm reveals that the results still hold in the absence of upward navigation. Thus, we have the following.

Proposition 6.16. *The weakly downward positive XPath algebra and the weakly downward core positive XPath algebra are both equivalent to* $\mathcal{X}(\downarrow, \pi_1, \pi_2)$.

We now summarize the characterization results.

Theorem 6.17. *Consider the weakly downward (core) positive XPath algebra. Let* $D = (V, Ed, r, \lambda)$ *be a document, and let* v_1 *and* v_2 *be nodes of* D*. Then* $v_1 \equiv_{\exp} v_2$ *if and only if* $v_1 \approxeq_{\updownarrow} v_2$.

Theorem 6.18. *Consider the weakly downward (core) positive XPath algebra. Let* $D = (V, Ed, r, \lambda)$ *be a document, and let* v_1*,* w_1*,* v_2*, and* w_2 *be nodes of* D *such that* w_1 *is a descendant of* v_1 *and* w_2 *is a descendant of* v_2*. Then, the property that, for each expression* e *in the language under consideration,* $(v_1, w_1) \in e(D)$ *if and only if* $(v_2, w_2) \in e(D)$ *is equivalent to the property* $(v_1, w_1) \cong_{\approxeq_{\updownarrow}} (v_2, w_2)$.

Theorem 6.19. *Let* $D = (V, Ed, r, \lambda)$ *be a document, and let* $R \subseteq V \times V$*. Then, there exists an expression* e *in the weakly downward (core) positive XPath algebra such that* $e(D) = R$ *if and only if,*

1. *for all* $v, w \in V$*,* $(v, w) \in R$ *implies* w *is a descendant of* v*; and,*
2. *for all* $v_1, w_1, v_2, w_2 \in V$ *with* w_1 *a descendant of* v_1*,* w_2 *a descendant of* v_2*, and* $(v_1, w_1) \cong_{\geq_{\updownarrow}} (v_2, w_2)$*,* $(v_1, w_1) \in R$ *implies* $(v_2, w_2) \in R$.

Corollary 6.20. *Let* $D = (V, Ed, r, \lambda)$ *be a document, let* v *be a node of* D*, and let* $W \subseteq V$*. Then there exists an expression* e *in the weakly downward (core) positive XPath algebra such that* $e(D)(v) = W$ *if and only if all nodes of* W *are descendants of* v*, and, for all nodes* w_1 *and* w_2 *of* D *with* $(v, w_1) \cong_{\geq_{\updownarrow}} (v, w_2)$*,* $w_1 \in W$ *implies* $w_2 \in W$.

Corollary 6.21. *Let* $D = (V, Ed, r, \lambda)$ *be a document, and let* $W \subseteq V$*. Then there exists an expression* e *in the weakly downward (core) positive XPath algebra such that* $e(D)(r) = W$ *if and only if, for all nodes* w_1 *and* w_2 *of* D *with* $w_1 \geq_{\updownarrow} w_2$*,* $w_1 \in W$ *implies* $w_2 \in W$.

Hence, the weakly downward positive (core) XPath algebra and the strictly downward positive (core) XPath algebra are navigationally equivalent if navigation always starts from the root.

7. Upward languages

In analogy to downward languages, we call a language *upward* if, for any expression in that language, and for any node v of the document D under consideration, all nodes in $e(D)(v)$ are ancestors of v. If in an addition, it is always the case that $e(D)(v) = e(D')$, where D' is the subtree of D obtained by removing from D all strict descendants of v, we call the language *strictly upward*. Upward languages that are *not* strictly upward will be called *weakly upward*.

For E a set of nonbasic operations of Table 1, $\mathcal{X}(E)$ or $\mathcal{C}(E)$ is upward if it does not contain downward navigation ("$\downarrow$"), and inverse ("$.^{-1}$"). Additionally, strictly upward languages do not contain second projection ("π_2") and counting operations ("$\mathrm{ch}_{\geq k}(.)$").

Of course, there is a distinct asymmetry between strictly upward languages and strictly downward languages: while a node can have an arbitrary number of children, it has at most one parent, making the analysis of strictly upward languages much easier than the analysis of downward languages. We shall see, however, that this asymmetry disappears for weakly upward languages versus weakly downward languages.

Finally, we observe that the analogues of Theorem 5.19 and Corollary 5.20 still hold for upward languages: set difference ("$-$") and intersection ("$\cap$") can be eliminated, unless they are used as operations in a Boolean combination of subexpressions of the language within a projection. Hence, an upward language and its corresponding core language coincide.

7.1. Strictly upward languages

The languages we consider here, are $\mathcal{X}(\uparrow, \pi_1, -)$ and $\mathcal{C}(\uparrow, \pi_1, -)$, which are equivalent, and $\mathcal{X}(\uparrow, \pi_1, \cap)$ and $\mathcal{C}(\uparrow, \pi_1, \cap)$, which are also equivalent. We refer to the former as the *strictly upward (core) XPath algebra* and the *strictly upward (core) positive XPath algebra*, respectively. As the characterization results for these languages are easy to derive along the lines set out in Section 5, we merely summarize the results.

Theorem 7.1. *Consider the strictly upward (core) (positive) XPath algebra. Let $D = (V, Ed, r, \lambda)$ be a document, and let v_1 and v_2 be nodes of D. Then $v_1 \equiv_{\exp} v_2$, if and only if $v_1 \equiv_{\uparrow} v_2$.*

Theorem 7.2. *Consider the strictly upward (core) (positive) XPath algebra. Let $D = (V, Ed, r, \lambda)$ be a document, and let v_1, w_1, v_2, and w_2 be nodes of D such that w_1 is an ancestor of v_1 and w_2 is an ancestor of v_2. Then, the property that, for each expression e in the language under consideration, $(v_1, w_1) \in e(D)$ if and only if $(v_2, w_2) \in e(D)$ is equivalent to the property $(v_1, w_1) \cong_{\equiv_{\uparrow}} (v_2, w_2)$.*

Theorem 7.3. *Let $D = (V, Ed, r, \lambda)$ be a document, and let $R \subseteq V \times V$. Then, there exists an expression e in the strictly upward (core) XPath algebra such that $e(D) = R$ if and only if,*

1. *for all $v, w \in V$, $(v, w) \in R$ implies w is a ancestor of v; and,*

2. *for all* $v_1, w_1, v_2, w_2 \in V$ *with* w_1 *a ancestor of* v_1*,* w_2 *a ancestor of* v_2*, and* $(v_1, w_1) \cong_{\equiv_\uparrow} (v_2, w_2)$*,* $(v_1, w_1) \in R$ *implies* $(v_2, w_2) \in R$*.*

Theorem 7.4. *Let* $D = (V, Ed, r, \lambda)$ *be a document, and let* $R \subseteq V \times V$*. Then, there exists an expression* e *in the strictly upward (core) positive XPath algebra such that* $e(D) = R$ *if and only if,*

1. *for all* $v, w \in V$*,* $(v, w) \in R$ *implies* w *is a ancestor of* v*; and,*
2. *for all* $v_1, w_1, v_2, w_2 \in V$ *with* w_1 *a ancestor of* v_1*,* w_2 *a ancestor of* v_2*, and* $(v_1, w_1) \cong_{\geq_\uparrow} (v_2, w_2)$*,* $(v_1, w_1) \in R$ *implies* $(v_2, w_2) \in R$*.*

The difference between the strictly downward (core) XPath algebra and the strictly downward (core) positive XPath algebra becomes only apparent in the BP-characterization: in Theorem 7.3, R is a union of equivalence classes under $\cong_{\equiv_\uparrow}$, whereas in Theorem 7.4, R is merely closed under the relation $\cong_{\geq_\uparrow}$.

7.2. Weakly upward languages

Weakly upward languages are closely related to weakly downward languages, by the following result.

Theorem 7.5. *Let* E *be a set of nonbasic operations not containing downward navigation ("*$\downarrow$*"), and inverse ("*$.^{-1}$*"). Let* E' *be the set of nonbasic operations obtained from* E *by replacing downward navigation by upward navigation ("*$\uparrow$*"), first projection ("*π_1*") by second projection ("*π_2*"), and second projection by first projection. Then, for each expression* e *in* $\mathcal{X}(E)$ *(respectively,* $\mathcal{C}(E)$*), there is an expression* e' *in* $\mathcal{X}(E')$ *(respectively,* $\mathcal{C}(E')$*) such that* e^{-1} *and* e' *are equivalent at the level of queries, and vice versa.*

Proof. Starting from e^{-1}, we eliminate inverse ("$.^{-1}$") using the identities in the proof of Proposition 2.3, and the additional identities $\pi_1(e)^{-1}(D) = \pi_2(e^{-1})(D)$ and $\pi_2^{-1}(D) = \pi_1(e^{-1})(D)$, for D an arbitrary document.[8] This elimination process yields the desired expression e'. □

Together with the fact that, in a subsumption or congruence, the order of the nodes in the pairs on the left- and right-hand sides may be swapped simultaneously (Proposition 3.5, (2) and (4)), Theorem 7.5 has the following immediate consequences:

1. Each characterization for a weakly downward language in Section 6—which in each instance contains both projections—yields a characterization for the corresponding weakly upward language (i.e., obtained by substituting upward navigation for downward navigation) by replacing "descendant" by "ancestor"; and

[8] Observe that $\pi_1(e)^{-1}(D) = \pi_1(e)(D)$ and $\pi_2(e)^{-1}(D) = \pi_2(e)(D)$ are also valid identities if the sole purpose was to eliminate inverse; however, these identities will not lead to the desired result.

2. Each characterization for a strictly downward language in Section 5—which in each instance contains the first projection—yields a characterization for the corresponding weakly upward language (i.e., obtained by substituting upward navigation for downward navigation and second for first projection) by replacing "descendant" by "ancestor".

Moreover, Theorem 7.5 gives us for free characterizations for some additional weakly *downward* languages *not* considered in Section 6:

> Each characterization for a strictly upward language in Section 7.1—which in each instance contains the second projection—yields a characterization for the corresponding weakly downward language (i.e., obtained by substituting downward navigation for downward navigation and first for second projection) by replacing "ancestor" by "descendant".

In view of space considerations, however, we refrain from explicitly writing down these new characterization results.

8. Languages for two-way navigation

We finally consider languages which are neither downward nor upward, i.e., in which navigation in both directions ("$\downarrow$" and "$\uparrow$") is possible. A notable difference in this case is that standard languages no longer always coincide with their associated core languages in expressive power. Below we distinguish languages with and without difference. In the first case, we discuss the standard languages and the core languages separately (Sections 8.1 and 8.2). In the second case, there is no need for this distinction (Section 8.3).

8.1. Standard languages with difference for two-way navigation

First, we state analogues to Lemmas 6.1 and 6.2 for pairs of nodes that are *not* necessarily ancestor-descendant pairs.

Lemma 8.1. *Let $D = (V, Ed, r, \lambda)$ be a document, let v_1, w_1, v_2, and w_2 be nodes of D, and let $k \geq 1$. Then, $(v_1, w_1) \cong_{\equiv^k_\updownarrow} (v_2, w_2)$ if and only if $(v_1, w_1) \cong (v_2, w_2)$, $v_1 \equiv^k_\updownarrow v_2$, and $w_1 \equiv^k_\updownarrow w_2$.*

Proof. As the "only if" is obvious, we focus on the "if." Obviously, $(v_1, w_1) \cong (v_2, w_2)$ implies that $(\mathrm{top}(v_1, w_1), v_1) \cong (\mathrm{top}(v_2, w_2), w_2)$. By Lemma 6.1, $(\mathrm{top}(v_1, w_1), v_1) \cong_{\equiv^k_\updownarrow} (\mathrm{top}(v_2, w_2), v_2)$. In the same way, we derive $(\mathrm{top}(v_1, w_1), w_1) \cong_{\equiv^k_\updownarrow} (\mathrm{top}(v_2, w_2), w_2)$. Applying Proposition 3.5, (2), (3) and (4), yields the desired result. $\square$

Lemma 8.2. *Let $D = (V, Ed, r, \lambda)$ be a document, let v_1 and w_1 be nodes of D, and let $k \geq 2$. Then,*

1. *for each node v_2 of D for which $v_1 \equiv^k_\updownarrow v_2$ there is a node w_2 in D such that $(v_1, w_1) \cong_{\equiv^k_\updownarrow} (v_2, w_2)$; and*

2. *for each node w_2 of D for which $w_1 \equiv_\updownarrow^k w_2$ there is a node v_2 in D such that $(v_1, w_1) \cong_{\equiv_\updownarrow^k} (v_2, w_2)$.*

Proof. We only prove (1); the proof of (2) is completely analogous. By Lemma 6.2, (2), there exists a node t_2 in D such that $(\mathrm{top}(v_1, w_1), v_1) \cong_{\equiv_\updownarrow^k} (t_2, v_2)$, and, hence, also that $(v_1, \mathrm{top}(v_1, w_1)) \cong_{\equiv_\updownarrow^k} (v_2, t_2)$, by Proposition 3.5, (2) and (4). Let y_1 be the child of $\mathrm{top}(v_1, w_1)$ on the path to w_1. Since $k \geq 2$, there is a child y_2 of t_2 such that (1) $y_1 \equiv_\updownarrow^k y_2$ and (2) y_2 is not on the path from t_2 to v_2.[9] By Lemma 6.2, (2), there exists a node w_2 in D such that $(y_1, w_1) \cong_{\equiv_\updownarrow^k} (y_2, w_2)$. In particular, $w_1 \equiv_\updownarrow^k w_2$. By construction, $t_2 = \mathrm{top}(v_2, w_2)$, and, hence, $(v_1, w_1) \cong (v_2, w_2)$. The result now follows from Lemma 8.1. $\square$

The mutual position of the nodes in the statement and the proof of Lemma 8.2, (1), is illustrated in Figure 5.

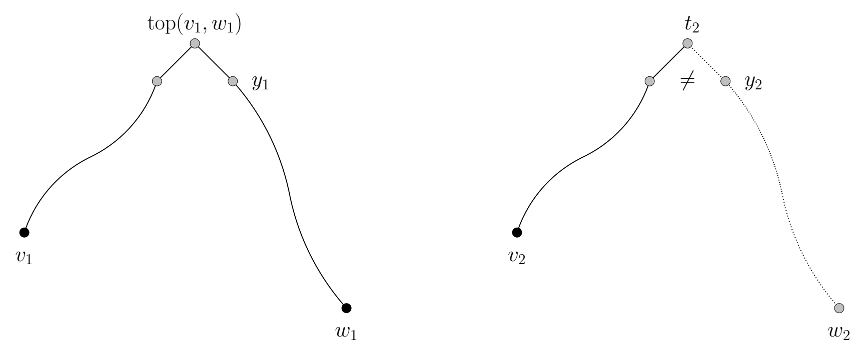

Figure 5: Mutual position of the nodes in the statement and the proof of Lemma 8.2, (1).

Before we can start with establishing the relationship between $\equiv_\updownarrow^k$-congruence and expression equivalence under languages allowing two-way navigation, we need one more lemma to be able to deal with the composition operator.

Lemma 8.3. *Let $D = (V, Ed, r, \lambda)$ be a document, let v_1, w_1, v_2, and w_2 be nodes of D such that $(v_1, w_1) \cong_{\equiv_\updownarrow^k} (v_2, w_2)$, and let $k \geq 3$. Then, for every node y_1 of D, there exists a node y_2 of D such that $(v_1, y_1) \cong_{\equiv_\updownarrow^k} (v_2, y_2)$, and $(y_1, w_1) \cong_{\equiv_\updownarrow^k} (y_2, w_2)$.*

Proof. The proof is essentially a case analysis. In each case description, we assume implicitly that the cases that were already dealt with before are excluded.

---

[9]To see the latter claim, observe that t_2 must have two different k-equivalent children when $\mathrm{top}(v_1, w_1)$ has.

1. y_1 *is on the path from* v_1 *to* w_1. In that case, let y_2 be the node corresponding to y_1 on the path from v_2 to w_2. The result now follows immediately.
2. y_1 *is a strict descendant of* v_1. By Lemma 6.2, (1), there is a (strict) descendant y_2 of v_2 such that $(v_1, y_1) \cong_{\equiv^k_\updownarrow} (v_2, y_2)$. The result now follows immediately.
3. y_1 *is a strict descendant of* w_1. Analogous to the previous case.
4. y_1 *is a strict ancestor of top*(v_1, w_1). By Lemma 6.2, (2), there is a (strict) ancestor y_2 of $\mathrm{top}(v_2, w_2)$ such that $(\mathrm{top}(v_1, w_1), y_1) \cong_{\equiv^k_\updownarrow} (\mathrm{top}(v_2, w_2), y_2)$. The result now follows immediately.
5. *top*(v_1, y_1) *is an internal node on the path from* v_1 *to top*(v_1, w_1). By Lemma 8.2, (1), there exists a node y_2 in D such that $(v_1, y_1) \cong_{\equiv^k_\updownarrow} (v_2, y_2)$. Since, in this case, $\mathrm{top}(y_1, w_1) = \mathrm{top}(v_1, w_1)$, and, therefore, an ancestor of v_1, we may apply Proposition 3.5, (2)–(4), to obtain that $(y_1, w_1) \cong (y_2, w_2)$. Since, moreover, $y_1 \equiv^k_\updownarrow y_2$ and $w_1 \equiv^k_\updownarrow w_2$, the desired result now follows from Lemma 8.1.
   Figure 6 illustrates this case and the constructions therein.

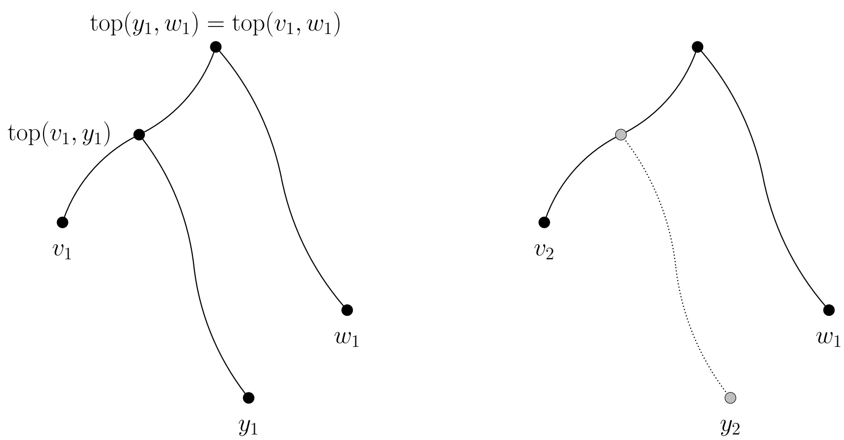

Figure 6: Mutual position of the nodes in Case 5 of the proof of Lemma 8.3.

6. *top*(y_1, w_1) *is an internal node on the path from top*(v_1, w_1) *to* w_1. Analogous to the previous case.
7. *top*$(v_1, y_1) =$ *top*(y_1, w_1) *is a strict ancestor of top*(v_1, w_1). By Lemma 8.2, (1), there exists a node y_2 in D such that $(v_1, y_1) \cong_{\equiv^k_\updownarrow} (v_2, y_2)$. Since, in this case, $\mathrm{top}(y_1, w_1)$ is a (strict) ancestor of $\mathrm{top}(v_1, w_1)$, and, therefore, an ancestor of v_1, we may apply Proposition 3.5, (2)–(4), to obtain that $(y_1, w_1) \cong (y_2, w_2)$. Since, moreover, $y_1 \equiv^k_\updownarrow y_2$ and $w_1 \equiv^k_\updownarrow w_2$, the desired result now follows from Lemma 8.1.
   Figure 7 illustrates this case and the constructions therein.

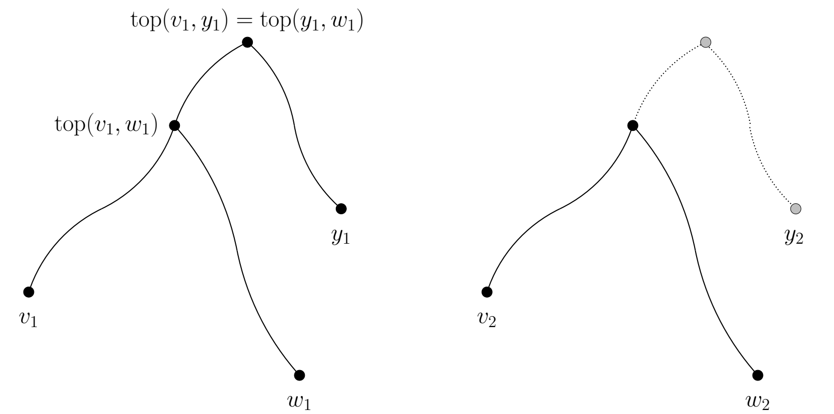

Figure 7: Mutual position of the nodes in Case 7 of the proof of Lemma 8.3.

8. *top*(v_1, y_1) = *top*(y_1, w_1) = *top*(v_1, w_1). Let z_1 be the child of the top node on the path to y_1. By assumption, z_1 is not on the path from v_1 to w_1. Since $k \geq 3$, there is a node z_2 in D not on the path from v_2 to w_2 such that $z_1 \equiv_{\updownarrow}^{k} z_2$. (For example, in the subcase where the children of top(v_1, w_1) on the paths to v_1, w_1, and y_1, the last of which is z_1, are all three k-equivalent, we know that top(v_2, w_2) must also have at least three children that are k-equivalent to z_1. Hence, at least one of these is not on the path from v_1 to w_1.) By Lemma 8.2, (1), there exists a node y_2 in D such that $(z_1, y_1) \cong_{\equiv_{\updownarrow}^{k}} (z_2, y_2)$. The result now follows readily.

   Figure 8 illustrates this case and the constructions therein.

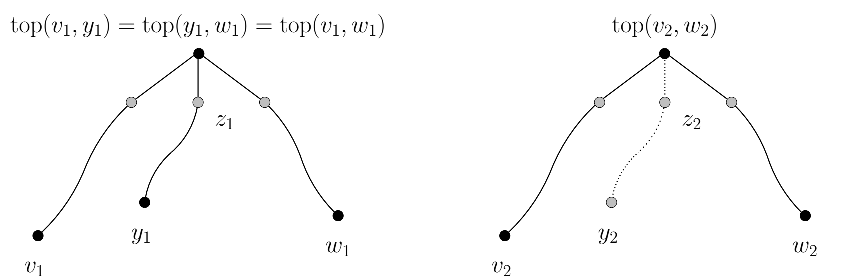

Figure 8: Mutual position of the nodes in Case 8 of the proof of Lemma 8.3.

□

We are now ready to state the analogue of Proposition 6.3 for languages with two-way navigation.

Proposition 8.4. *Let $k \geq 3$, and let E be the set of all nonbasic operations in Table 1, except for selection on at least m children satisfying some condition ("$\mathrm{ch}_{\geq m}(.)$") for $m > k$. Let e be an expression in $\mathcal{X}(E)$. Let $D = (V, Ed, r, \lambda)$ be a document, and let v_1, w_1, v_2, and w_2 be nodes of D. Assume furthermore that $(v_1, w_1) \cong_{\equiv^k_\updownarrow} (v_2, w_2)$. Then, $(v_1, w_1) \in e(D)$ if and only if $(v_2, w_2) \in e(D)$.*

Proof. The proof goes along the same lines as the proofs of Propositions 6.3 and 5.2. The base case, for the atomic operators, remains straightforward. In the induction step, we must now rely on Lemma 8.3 to make the case for composition ("./."). To make the case for first, respectively, second projection ("π_1," respectively, "π_2"), we must rely on Lemma 8.2, (1), respectively, (2). The arguments for the counting operations ("$\mathrm{ch}_{\geq m}(.)$," $m \leq k$), union ("$\cup$"), intersection ("$\cap$"), and set difference ("$-$") in the proof of Proposition 5.2 carry over to the present setting. Finally, the case for inverse ("$.^{-1}$") is straightforward. □

As in Section 6.2, we can in two steps infer the following result from Proposition 8.4.

Corollary 8.5. *Let $k \geq 3$, and let E be a set of nonbasic operations not containing selection on at least m children satisfying some condition ("$\mathrm{ch}_{\geq m}(.)$") for $m > k$. Consider the language $\mathcal{X}(E)$. Let $D = (V, Ed, r, \lambda)$ be a document, and let v_1 and v_2 be nodes of D. If $v_1 \equiv^k_\updownarrow v_2$, then $v_1 \equiv_{\exp} v_2$.*

Now, notice that Proposition 6.6 of Section 6.2 is also applicable to an important class of languages allowing two-way navigation. The standard language for two-way navigation satisfying both Corollary 8.5 and Proposition 6.6 is $\mathcal{X}(\downarrow, \uparrow, \mathrm{ch}_{\geq 1}(.), \ldots, \mathrm{ch}_{\geq k}(.), -)$.[10] We call this language the *XPath algebra with counting up to k*. Combining the aforementioned results, we obtain the following.

Theorem 8.6. *Let $k \geq 3$, and consider the XPath algebra with counting up to k. Let $D = (V, Ed, r, \lambda)$ be a document, and let v_1 and v_2 be nodes of D. Then, $v_1 \equiv_{\exp} v_2$ if and only if $v_1 \equiv^k_\updownarrow v_2$.*

By Proposition 2.4, selection on up to three children satisfying some condition ("$\mathrm{ch}_{\geq m}(.)$," $1 \leq m \leq 3$) can be expressed in the XPath algebra. Hence, a special case arises for $k = 3$:

Corollary 8.7. *Consider the XPath algebra. Let $D = (V, Ed, r, \lambda)$ be a document, and let v_1 and v_2 be nodes of D. Then, $v_1 \equiv_{\exp} v_2$ if and only if $v_1 \equiv^3_\updownarrow v_2$.*

We next prove a converse to Proposition 8.4.

Proposition 8.8. *Let $k \geq 3$, and consider the XPath algebra with counting up to k. Let $D = (V, Ed, r, \lambda)$ be a document, and let v_1, w_1, v_2, and w_2 be nodes of D. Assume furthermore that, for each expression e in the language, $(v_1, w_1) \in e(D)$ if and only if $(v_2, w_2) \in e(D)$. Then $(v_1, w_1) \cong_{\equiv^k_\updownarrow} (v_2, w_2)$.*

[10] All other operations are redundant, by Proposition 2.3.

Proof. Since $\mathrm{sig}(v_1, w_1)$ is an expression in the language under consideration, and since $(v_1, w_1) \in \mathrm{sig}(v_1, w_1)$, $(v_2, w_2) \in \mathrm{sig}(v_1, w_1)$. Similarly, $(v_1, w_1) \in \mathrm{sig}(v_2, w_2)$. We may thus conclude that $(v_1, w_1) \cong (v_2, w_2)$. Now, let f be any expression in the language such that $f(D)(v_1) \neq \emptyset$. Then, $(v_1, v_1) \in \pi_1(f)(D)$. Let $e := \pi_1(f)/\mathrm{sig}(v_1, w_1)$. By construction, $(v_1, w_1) \in e(D)$. Hence, by assumption, $(v_2, w_2) \in e(D)$, which implies $(v_2, v_2) \in \pi_1(f)(D)$ or $f(D)(v_2) \neq \emptyset$. The same holds vice versa, and we may thus conclude that $v_1 \equiv_{\mathrm{exp}} v_2$, and, hence, by Theorem 8.6, $v_1 \equiv^k_\updownarrow v_2$. In a similar way, we prove that $w_1 \equiv^k_\updownarrow w_2$. By Lemma 8.1, we may now conclude that $(v_1, w_1) \cong_{\equiv^k_\updownarrow} (v_2, w_2)$. □

Combining Propositions 8.4 and 8.8, we obtain the following characterization.

Corollary 8.9. *Let $k \geq 3$, and consider the XPath algebra with counting up to k. Let $D = (V, Ed, r, \lambda)$ be a document, and let v_1, w_1, v_2, and w_2 be nodes of D. Then, the property that, for each expression e in the language under consideration, $(v_1, w_1) \in e(D)$ if and only if $(v_2, w_2) \in e(D)$ is equivalent to the property $(v_1, w_1) \cong_{\equiv^k_\updownarrow} (v_2, w_2)$.*

Using Theorem 8.6 instead of Theorem 5.7, we can recast the proof of Lemma 5.11 into a proof of

Lemma 8.10. *Let $k \geq 3$. Let $D = (V, Ed, r, \lambda)$ be a document, and let v_1 be a node of D. There exists an expression e_{v_1} in the XPath algebra with counting up to k such that, for each node v_2 of D, $e_{v_1}(D)(v_2) \neq \emptyset$ if and only if $v_1 \equiv^k_\updownarrow v_2$.*

We can now bootstrap Lemma 8.10 to the following result.

Lemma 8.11. *Let $k \geq 3$. Let $D = (V, Ed, r, \lambda)$ be a document, and let v_1 and w_1 be a nodes of D. There exists an expression e_{v_1,w_1} in the XPath algebra with counting up to k such that, for all nodes v_2 and w_2 of D, $(v_2, w_2) \in e_{v_1,w_1}(D)$ if and only if $(v_1, w_1) \cong_{\equiv^k_\updownarrow} (v_2, w_2)$.*

Proof. From Lemma 8.10, we know that, for node y_1 of D, there exists an expression e_{y_1} in the XPath algebra with counting up to k such that, for each node y_2 of D, $e_{y_1}(D)(y_2) \neq \emptyset$ if and only if $y_1 \equiv^k_\updownarrow y_2$. Now, let v_1 and w_1 be nodes of D. Let $\mathrm{sig}(v_1, w_1) = \uparrow^u/\downarrow^d$, with $u, d \geq 0$, and define

$$e_{v_1,w_1} := \pi_1(e_{v_1})/\mathrm{sig}(v_1, w_1)/\pi_1(e_{w_1}) - \uparrow^{u-1}/\downarrow^{d-1},$$

where, for an expression f, we define $f^{-1} := \emptyset$. Clearly, e_{v_1,w_1} is also in the XPath algebra with counting up to k. Let v_2 and w_2 be nodes of D. Suppose $(v_2, w_2) \in e_{v_1,w_1}(D)$. Then, by Proposition 3.4, $\mathrm{sig}(v_1, w_1) = \mathrm{sig}(v_2, w_2)$. Furthermore, it follows that $(v_2, v_2) \in e_{v_1}(D)$ and $(w_2, w_2) \in e_{w_1}(D)$. By Lemma 8.10, $v_1 \equiv^k_\updownarrow v_2$ and $w_1 \equiv^k_\updownarrow w_2$. It now follows from Lemma 8.1 that $(v_1, w_2) \cong_{\equiv^k_\updownarrow} (v_2, w_2)$. As $(v_1, w_1) \in e_{v_1,w_1}(D)$, the converse follows from Corollary 8.9. □

The BP characterization results now follow readily.

Theorem 8.12. *Let $k \geq 3$. Let $D = (V, Ed, r, \lambda)$ be a document, and let $R \subseteq V \times V$. Then, there exists an expression e in the XPath algebra with counting up to k such that $e(D) = R$ if and only if, for all $v_1, w_1, v_2, w_2 \in V$ with $(v_1, w_1) \cong_{\equiv^k_\updownarrow} (v_2, w_2)$, $(v_1, w_1) \in R$ implies $(v_2, w_2) \in R$.*

The specialization to the XPath algebra is as follows.

Corollary 8.13. *Let $D = (V, Ed, r, \lambda)$ be a document, and let $R \subseteq V \times V$. There exists an expression e in the XPath algebra such that $e(D) = R$ if and only if, for all $v_1, w_1, v_2, w_2 \in V$ with $(v_1, w_1) \cong_{\equiv^3_\updownarrow} (v_2, w_2)$, $(v_1, w_1) \in R$ implies $(v_2, w_2) \in R$.*

We recast Theorem 8.12 and Corollary 8.13 in terms of node-level navigation.

Theorem 8.14. *Let $k \geq 3$. Let $D = (V, Ed, r, \lambda)$ be a document, let v be a node of D, and let $W \subseteq V$. Then there exists an expression e in the XPath algebra with counting up to k such that $e(D)(v) = W$ if and only if, for all $w_1, w_2 \in W$ with $(v, w_1) \cong_{\equiv^k_\updownarrow} (v, w_2)$, $w_1 \in W$ implies $w_2 \in W$.*

The specialization to the XPath algebra is as follows.

Corollary 8.15. *Let $D = (V, Ed, r, \lambda)$ be a document, let v be a node of D, and let $W \subseteq V$. Then there exists an expression e in the XPath algebra such that $e(D)(v) = W$ if and only if, for all $w_1, w_2 \in W$ with $(v, w_1) \cong_{\equiv^3_\updownarrow} (v, w_2)$, $w_1 \in W$ implies $w_2 \in W$.*

Finally, we consider the special case where navigation starts from the root. For $v = r$, the condition $(v, w_1) \cong_{\equiv^k_\updownarrow} (v, w_2)$ reduces to $w_1 \equiv^k_\updownarrow w_2$, by Proposition 4.17 and Lemma 6.1. Comparing Theorem 8.14 and Corollary 8.15 with, respectively, Theorem 5.17 and Corollary 5.18 then immediately yields the following.

Theorem 8.16. *Let $D = (V, Ed, r, \lambda)$.*

1. *for each expression e in the XPath algebra with counting up to k, $k \geq 3$, there exists an expression e' in the strictly downward (core) XPath algebra with counting up to k such that $e(D)(r) = e'(D)(r)$.*
2. *for each expression e in the XPath algebra, there exists an expression e' in the strictly downward (core) XPath algebra with counting up to 3 such that $e(D)(r) = e'(D)(r)$.*

Theorem 8.16 extends Theorem 6.15. When navigating from the root, the only thing that the full XPath algebra adds compared to using the strictly downward (core) XPath algebra is its ability to select on at least 2 and on at least 3 children satisfying some condition.

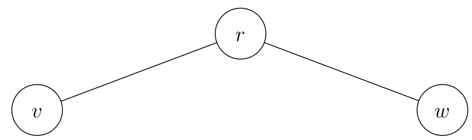

Figure 9: Document of Example 8.17.

8.2. Core languages with difference for two-way navigation

We now investigate what changes if we replace a standard language with difference for two-way navigation by the corresponding core language. The most important observation is that both languages are *not* equivalent, unlike in the cases of downward or upward navigation.

Example 8.17. Let $D = (V, Ed, r, \lambda)$ be the very simple document in Figure 9. For every value of $k \geq 2$,[11] $e := \uparrow/\downarrow - \varepsilon$ is an expression in the XPath algebra with counting up to k. We have that $e(D) = \{(v, w), (w, v)\}$. From Proposition 8.19, it will follow, however, that, for every expression e' in the corresponding core language, $(v, w) \in e'(D)$ implies that not only $(w, v) \in e'(D)$, but also $(v, v) \in e'(D)$ and $(w, w) \in e'(D)$.

We now explore which changes occur when we try to make the same reasoning as in Section 8.1.

As Example 8.17 suggests, there is no hope that we can express congruence in the core XPath algebra with counting up to k,[12] for any $k \geq 2$. Therefore we shall have to work with subsumption instead of congruence.

Lemma 8.1 still holds if we replace congruence by subsumption. We may of course still use Lemma 8.2 (as replacing congruence by subsumption here would yield a weaker statement). Lemma 8.3 also survives replacing congruence by subsumption, except that we can then strengthen its statement, as follows.

Lemma 8.18. *Let $D = (V, Ed, r, \lambda)$ be a document, let v_1, w_1, v_2, and w_2 be nodes of D such that $(v_1, w_1) \gtrsim_{\equiv^k_\updownarrow} (v_2, w_2)$, and let $k \geq 2$. Then, for every node y_1 of D, there exists a node y_2 of D such that $(v_1, y_1) \gtrsim_{\equiv^k_\updownarrow} (v_2, y_2)$, and $(y_1, w_1) \gtrsim_{\equiv^k_\updownarrow} (y_2, w_2)$.*

Proof. The only case in the proof of Lemma 8.3 where we used $k \geq 3$ is Case 8 ($\mathrm{top}(v_1, y_1) = \mathrm{top}(y_1, w_1) = \mathrm{top}(v_1, w_1)$) to guarantee that the path from $\mathrm{top}(v_2, w_2)$ to y_2 has no overlap with both the path from $\mathrm{top}(v_2, w_2)$ to v_2 and the path from $\mathrm{top}(v_2, w_2)$ to w_2. As this is no concern anymore when we

[11] We will not consider $k = 1$, because both $\mathrm{ch}_{\geq 1}(.)$ and $\mathrm{ch}_{\geq 2}(.)$ can be expressed in the core XPath algebra, by Proposition 2.4.

[12] This is the name we give to the core language corresponding to the (standard) XPath algebra with counting up to k

consider subsumption rather than congruence, the condition $k \geq 2$ suffices to recast the proof of Lemma 8.3 into a proof of Lemma 8.18. $\square$

Lemma 8.3 was used to complete the induction step for composition ("/") in the proof of Proposition 8.4. If we replace Lemma 8.3 by Lemma 8.18, we can also avoid making the assumption ≥ 3 here. Thanks to the restricted use of difference in core languages, we can also get away with subsumption instead of congruence.

Proposition 8.19. *Let $k \geq 2$, and let E be the set of all nonbasic operations in Table 1, except for selection on at least m children satisfying some condition ("$ch_{\geq m}(.)$") for $m > k$. Let e be an expression in $\mathcal{C}(E)$. Let $D = (V, Ed, r, \lambda)$ be a document, and let v_1, w_1, v_2, and w_2 be nodes of D. Assume furthermore that $(v_1, w_1) \gtrsim_{\equiv^k_\updownarrow} (v_2, w_2)$. Then, $(v_1, w_1) \in e(D)$ implies $(v_2, w_2) \in e(D)$.*

Proof. The proof goes along the same lines as the proof of Proposition 8.4, except that, in the induction step, we need not consider the case of set difference ("$-$"). However, we must consider instead the case where the expression is of the form $e := \pi_1(f)$ or $e := \pi_2(f)$ with f a Boolean combination of expressions of $\mathcal{C}(E)$ satisfying the induction hypothesis. For reasons of symmetry, we only consider the case $e := \pi_1(f)$. Without loss of generality, we may assume that f is union-free. Indeed, we can always rewrite f in disjunctive normal form, and, for $f = f_1 \cup f_2$, $\pi_1(f) = \pi_1(f_1) \cup \pi_1(f_2)$. If, for an expression g in $\mathcal{C}(E)$, we define $\overline{g}$ by $\overline{g}(D) := V \times V - g(D)$, we can write $f = f_1 \cap \ldots f_p \cap \overline{g_1} \cap \ldots \overline{g_q}$ for some $p \geq 1$ and $q \geq 0$, with $f_1, \ldots, f_p, g_1, \ldots, g_q$ in $\mathcal{C}(E)$ and satisfying the induction hypothesis. In particular, if $(v_1, v_1) \in \pi_1(f)(D)$,[13] there exists a node y_1 in D such that $(v_1, y_1) \in f_1(D), \ldots, (v_1, y_1) \in f_p(D)$ and $(v_1, y_1) \notin g_1(D), \ldots, (v_1, y_1) \notin g_p(D)$. By Lemma 8.2, there exists a node y_2 in D such that $(v_1, y_1) \cong_{\equiv^k_\updownarrow} (v_2, y_2)$. Hence, $(v_1, y_1) \gtrsim_{\equiv^k_\updownarrow} (v_2, y_2)$ *and* $(v_2, y_2) \gtrsim_{\equiv^k_\updownarrow} (v_1, y_1)$. By the induction hypothesis, $(v_2, y_2) \in f_1(D), \ldots, (v_2, y_2) \in f_p(D)$. Now, assume that, for some j, $1 \leq j \leq q$, $(v_2, y_2) \in g_j(D)$. Then, again by the induction hypothesis, $(v_1, y_1) \in g_j(D)$, a contradiction. Hence, $(v_2, y_2) \notin g_1(D), \ldots, (v_2, y_2) \notin g_p(D)$. We may thus conclude that $(v_2, v_2) \in \pi_1(f)$. $\square$

By applying Proposition 8.19 twice, we obtain the following.

Corollary 8.20. *Let $k \geq 2$, and let E be the set of all nonbasic operations in Table 1, except for selection on at least m children satisfying some condition ("$ch_{\geq m}(.)$") for $m > k$. Let e be an expression in $\mathcal{C}(E)$. Let $D = (V, Ed, r, \lambda)$ be a document, and let v_1, w_1, v_2, and w_2 be nodes of D. Assume furthermore that $(v_1, w_1) \cong_{\equiv^k_\updownarrow} (v_2, w_2)$. Then, $(v_1, w_1) \in e(D)$ if and only if $(v_2, w_2) \in e(D)$.*

As in Section 6.2, we can in two steps infer the following result from Corollary 8.20.

[13] In this case, $v_1 = w_1$ and $v_2 = w_2$.

Corollary 8.21. *Let $k \geq 2$, and let E be a set of nonbasic operations not containing selection on at least m children satisfying some condition ("$\mathrm{ch}_{\geq m}(.)$") for $m > k$. Consider the language $\mathcal{C}(E)$. Let $D = (V, Ed, r, \lambda)$ be a document, and let v_1 and v_2 be nodes of D. If $v_1 \equiv^k_{\updownarrow} v_2$, then $v_1 \equiv_{\mathrm{exp}} v_2$.*

Notice that, for $k \geq 3$, Corollary 8.21 is also an immediate consequence of Corollary 8.5. Because we are dealing with a weaker language, we can also include the case $k = 2$, however.

We already observed that Proposition 6.6 of Section 6.2 is also applicable to an important class of languages allowing two-way navigation. The core language for two-way navigation satisfying both Corollary 8.21 and Proposition 6.6 is $\mathcal{C}(\downarrow, \uparrow, \pi_1, \pi_2, \mathrm{ch}_{\geq 1}(.), \ldots, \mathrm{ch}_{\geq k}(.), -)$.[14] We call this language the *core XPath algebra with counting up to k*. Combining the aforementioned results, we obtain the following.

Theorem 8.22. *Let $k \geq 2$, and consider the core XPath algebra with counting up to k. Let $D = (V, Ed, r, \lambda)$ be a document, and let v_1 and v_2 be nodes of D. Then, $v_1 \equiv_{\mathrm{exp}} v_2$ if and only if $v_1 \equiv^k_{\updownarrow} v_2$.*

By Proposition 2.4, selection on up to two children satisfying some condition ("$\mathrm{ch}_{\geq m}(.)$," $1 \leq m \leq 2$) can be expressed in the core XPath algebra. Hence, a special case arises for $k = 2$:

Corollary 8.23. *Consider the core XPath algebra. Let $D = (V, Ed, r, \lambda)$ be a document, and let v_1 and v_2 be nodes of D. Then, $v_1 \equiv_{\mathrm{exp}} v_2$ if and only if $v_1 \equiv^2_{\updownarrow} v_2$.*

The proof of Proposition 8.8 can be recast to a proof of the following converse to Proposition 8.19.

Proposition 8.24. *Let $k \geq 2$, and consider the core XPath algebra with counting up to k. Let $D = (V, Ed, r, \lambda)$ be a document, and let v_1, w_1, v_2, and w_2 be nodes of D. Assume furthermore that, for each expression e in the language, $(v_1, w_1) \in e(D)$ implies $(v_2, w_2) \in e(D)$. Then $(v_1, w_1) \gtrsim_{\equiv^k_{\updownarrow}} (v_2, w_2)$.*

Combining Propositions 8.19 and 8.24, we obtain the following characterization.

Corollary 8.25. *Let $k \geq 2$, and consider the core XPath algebra with counting up to k. Let $D = (V, Ed, r, \lambda)$ be a document, and let v_1, w_1, v_2, and w_2 be nodes of D. Then,*

1. *the property that, for each expression e in the language under consideration, $(v_1, w_1) \in e(D)$ implies $(v_2, w_2) \in e(D)$ is equivalent to the property $(v_1, w_1) \gtrsim_{\equiv^k_{\updownarrow}} (v_2, w_2)$; and,*

---

[14] Inverse ("$^{-1}$") is redundant, by the identities in the proof of Proposition 2.3, complemented by $\pi_1(e)^{-1}(D) = \pi_1(e)(D)$ and $\pi_2(e)^{-1}(D) = \pi_2(e)(D)$.

2. *the property that, for each expression e in the language under consideration, $(v_1, w_1) \in e(D)$ if and only if $(v_2, w_2) \in e(D)$ is equivalent to the property $(v_1, w_1) \cong_{\equiv^k_\updownarrow} (v_2, w_2)$.*

Lemma 8.10 also holds for the core XPath algebra (with the condition $k \geq 3$ replaced by $k \geq 2$). Lemma 8.11 is a different story, unfortunately. Example 8.17 already indicates that, given nodes v_1, w_1, v_2, and w_2 of a document D, we can in general not hope for an expression e_{v_1,w_1} such that $(v_2, w_2) \in e_{v_1,w_1}(D)$ if and only if $(v_1, w_1) \cong_{\equiv^k_\updownarrow} (v_2, w_2)$. The version with congruence replaced by subsumption does hold, however.

Lemma 8.26. *Let $k \geq 2$. Let $D = (V, Ed, r, \lambda)$ be a document, and let v_1 and w_1 be a nodes of D. There exists an expression e_{v_1,w_1} in the core XPath algebra with counting up to k such that, for all nodes v_2 and w_2 of D, $(v_2, w_2) \in e_{v_1,w_1}(D)$ if and only if $(v_1, w_1) \gtrsim_{\equiv^k_\updownarrow} (v_2, w_2)$.*

Proof. The proof follows the lines of the proof of Proposition 8.11 very closely, the main difference being that, from the proposed expression, the minus term must be omitted. □

The BP characterization results now follow readily.

Theorem 8.27. *Let $k \geq 2$. Let $D = (V, Ed, r, \lambda)$ be a document, and let $R \subseteq V \times V$. Then, there exists an expression e in the core XPath algebra with counting up to k such that $e(D) = R$ if and only if, for all $v_1, w_1, v_2, w_2 \in V$ with $(v_1, w_1) \gtrsim_{\equiv^k_\updownarrow} (v_2, w_2)$, $(v_1, w_1) \in R$ implies $(v_2, w_2) \in R$.*

The specialization to the core XPath algebra is as follows.

Corollary 8.28. *Let $D = (V, Ed, r, \lambda)$ be a document, and let $R \subseteq V \times V$. There exists an expression e in the core XPath algebra such that $e(D) = R$ if and only if, for all $v_1, w_1, v_2, w_2 \in V$ with $(v_1, w_1) \gtrsim_{\equiv^3_\updownarrow} (v_2, w_2)$, $(v_1, w_1) \in R$ implies $(v_2, w_2) \in R$.*

We recast Theorem 8.27 and Corollary 8.28 in terms of node-level navigation.

Theorem 8.29. *Let $k \geq 2$. Let $D = (V, Ed, r, \lambda)$ be a document, let v be a node of D, and let $W \subseteq V$. Then there exists an expression e in the core XPath algebra with counting up to k such that $e(D)(v) = W$ if and only if, for all $w_1, w_2 \in W$ with $(v, w_1) \gtrsim_{\equiv^k_\updownarrow} (v, w_2)$, $w_1 \in W$ implies $w_2 \in W$.*

The specialization to the core XPath algebra is as follows.

Corollary 8.30. *Let $D = (V, Ed, r, \lambda)$ be a document, let v be a node of D, and let $W \subseteq V$. Then there exists an expression e in the core XPath algebra such that $e(D)(v) = W$ if and only if, for all $w_1, w_2 \in W$ with $(v, w_1) \gtrsim_{\equiv^2_\updownarrow} (v, w_2)$, $w_1 \in W$ implies $w_2 \in W$.*

Finally, for the special case where navigation starts from the root, Theorem 8.29 and Corollary 8.30 reduce to the following.

Theorem 8.31. *Let $D = (V, Ed, r, \lambda)$.*

1. *for each expression e in the core XPath algebra with counting up to k, $k \geq 2$, there exists an expression e' in the strictly downward (core) XPath algebra with counting up to k such that $e(D)(r) = e'(D)(r)$.*
2. *for each expression e in the core XPath algebra, there exists an expression e' in the strictly downward (core) XPath algebra with counting up to 2 such that $e(D)(r) = e'(D)(r)$.*

Together with Theorem 8.16, Theorem 8.31 extends Theorem 6.15. When navigating from the root, the only thing that the core XPath algebra adds compared to using the strictly downward (core) XPath algebra is its ability to select on at least 2 children satisfying some condition.

8.3. Languages without difference for two-way navigation

As before with languages not containing difference, we do not consider counting operations, corresponding to considering the various syntactic notions of relatedness or equivalence between nodes only for the case $k = 1$. Taking into account Proposition 2.3, and recognizing that the techniques used in this paper to establish characterizations heavily use intersection, this means that only the following two languages must be considered:

- the language $\mathcal{X}(\downarrow, \uparrow, \cap)$, which we call the *positive XPath algebra*; and
- the language $\mathcal{C}(\downarrow, \uparrow, \pi_1, \pi_2, \cap)$, which we call the *core positive XPath algebra.*

Some of the present authors showed in [31] that $\mathcal{X}(\downarrow, \uparrow, \cap)$ and $\mathcal{X}(\downarrow, \uparrow, \pi_1, \pi_2)$ are equivalent in expressive power (even at the level of queries). Since obviously $\mathcal{X}(\downarrow, \uparrow, \pi_1, \pi_2) = \mathcal{C}(\downarrow, \uparrow, \pi_1, \pi_2)$, it follows readily that the positive XPath algebra and the core positive XPath algebra are equivalent.

The following results were already proved in [31], and are only repeated for completeness' sake.

Theorem 8.32. *Consider the (core) positive XPath algebra. Let $D = (V, Ed, r, \lambda)$ be a document, and let v_1 and v_2 be nodes of D. Then,*

1. *$v_1 \geq_{\mathrm{exp}} v_2$ if and only if $v_1 \geq^1_{\updownarrow} v_2$; and*
2. *$v_1 \equiv_{\mathrm{exp}} v_2$ if and only if $v_1 \approxeq_{\updownarrow} v_2$.*

Theorem 8.33. *Consider the (core) positive XPath algebra. Let $D = (V, Ed, r, \lambda)$ be a document, and let v_1, v_2, w_1, and w_2 be nodes of D. Then,*

1. *the property that, for each (core) positive XPath expression e, $(v_1, w_1) \in e(D)$ implies $(v_2, w_2) \in e(D)$ is equivalent to $(v_1, w_1) \gtrsim_{\geq^1_{\updownarrow}} (v_2, w_2)$; and*

2. *the property that, for each expression e in the language under consideration, $(v_1, w_1) \in e(D)$ if and only if $(v_2, w_2) \in e(D)$ is equivalent to the property $(v_1, w_1) \cong_{\approx_\updownarrow} (v_2, w_2)$.*

As in Section 6.5, we can bootstrap these results to BP-type characterizations.

Theorem 8.34. *Let $D = (V, Ed, r, \lambda)$ be a document, and let $R \subseteq V \times V$. Then, there exists an expression e in the (core) positive XPath algebra such that $e(D) = R$ if and only if, for all $v_1, w_1, v_2, w_2 \in V$ with $(v_1, w_1) \gtrsim_{\geq_\updownarrow} (v_2, w_2)$, $(v_1, w_1) \in R$ implies $(v_2, w_2) \in R$.*

Finally, Theorem 8.34 can be specialized to the node level, as follows.

Corollary 8.35. *Let $D = (V, Ed, r, \lambda)$ be a document, let v be a node of D, and let $W \subseteq V$. Then there exists an expression e in the (core) positive XPath algebra such that $e(D)(v) = W$ if and only if, for all nodes w_1 and w_2 of D with $(v, w_1) \gtrsim_{\geq_\updownarrow} (v, w_2)$, $w_1 \in W$ implies $w_2 \in W$.*

Corollary 8.36. *Let $D = (V, Ed, r, \lambda)$ be a document, and let $W \subseteq V$. Then there exists an expression e in the (core) positive XPath algebra such that $e(D)(r) = W$ if and only if, for all nodes w_1 and w_2 of D with $w_1 \geq_\updownarrow w_2$, $w_1 \in W$ implies $w_2 \in W$.*

Hence, the (core) positive XPath algebra, the weakly downward positive (core) XPath algebra, and the strictly downward positive (core) XPath algebra are all navigationally equivalent if navigation always starts from the root.

9. Discussion

In this paper, we characterized the expressive power of several natural fragments of XPath at the document level, as summarized in Table 5. Of course, it is possible to consider other fragments or extensions of the XPath algebra and its data model. Analyzing these using our two-step methodology in order to further improve our understanding of the instance expressivity of Tarski's algebra is one possible research direction which we have pursued recently [10, 11, 31].

Another future research direction is refining the links between XPath and finite-variable first-order logics [32]. Indeed, such links have been established at the level of query semantics. For example, Marx [33] has shown that an extended version of Core XPath is equivalent to $\mathrm{FO}^2_{\mathtt{tree}}$—first-order logic using at most two variables over *ordered* node-labeled trees—interpreted in the signature `child`, `descendant`, and `following_sibling`.

Our results establish new links to finite-variable first-order logics at the document level. For example, we can show that, on a given document, the XPath algebra and FO^3—first-order logic with at most three variables—are equivalent in expressive power. Indeed, as we discussed above, at the document level, the XPath-algebra is equivalent with Tarski's relation algebra [2] over

Table 5: Summary of main results.

Language	*Node relationship*	*Node Coupling Theorem*	*Path Relationship*	*Path Coupling Theorem*	*BP Result*
strictly downward (core) XPath algebra with counting up to k	$\equiv^k_\downarrow$	Theorem 5.7	$\cong_{\equiv^k_\downarrow}$	Corollary 5.10	Theorem 5.13
strictly downward (core) positive XPath algebra	$\approxeq_\downarrow$	Theorem 5.33	$\cong_{\approxeq_\downarrow}$	Theorem 5.34	Theorem 5.36
weakly downward (core) XPath algebra with counting up to k	$\equiv^k_\updownarrow$	Theorem 6.7	$\cong_{\equiv^k_\updownarrow}$	Corollary 6.10	Theorem 6.11
weakly downward (core) positive XPath algebra	$\approxeq_\updownarrow$	Theorem 6.17	$\cong_{\approxeq_\updownarrow}$	Theorem 6.18	Theorem 6.19
strictly upward (core) XPath algebra	$\equiv_\uparrow$	Theorem 7.1	$\cong_{\equiv_\uparrow}$	Theorem 7.2	Theorem 7.3
strictly upward (core) positive XPath algebra	$\equiv_\uparrow$	Theorem 7.1	$\cong_{\equiv_\uparrow}$	Theorem 7.2	Theorem 7.4
weakly upward languages			*see Section 7.2*		
XPath algebra with counting up to k	$\equiv^k_\updownarrow$	Theorem 8.6	$\cong_{\equiv^k_\updownarrow}$	Corollary 8.9	Theorem 8.12
core XPath algebra with counting up to k	$\equiv^k_\updownarrow$	Theorem 8.22	$\cong_{\equiv^k_\updownarrow}$	Corollary 8.25	Theorem 8.27
(core) positive XPath algebra ([31])	$\approxeq_\updownarrow$	Theorem 8.32	$\cong_{\approxeq_\updownarrow}$	Theorem 8.33	Theorem 8.34

trees. Tarski and Givant [4, 6] established the link between Tarski's algebra and FO^3. Corollary 8.7 can then be used to give a new characterization, other than via pebble-games [32, 34], of when two nodes in an unordered tree are indistinguishable in FO^3. In this light, connections between other fragments of the XPath algebra and finite-variable logics must be examined.

The connection between the XPath algebra and FO^3 also has ramifications with regard to complexity issues. Indeed, using a result of Grohe [35] which establishes that expression equivalence for FO^3 is decidable in polynomial time, it follows readily from Corollaries 8.13 and 8.15 that the global and local definability problems for the XPath algebra are decidable in polynomial time. Using the syntactic characterizations in this paper, one can also establish that the global and local definability problems for the other fragments of the XPath algebra are decidable in polynomial time. As mentioned in the Introduction, this feasibility suggests efficient partitioning and reduction techniques on the set of nodes and the set of paths in a document. Such techniques might be successfully applied towards various aspects of XML document processing such as indexing, access control, and document compression. This is another research direction which we are currently pursuing [12, 36].

References

[1] M. Gyssens, J. Paredaens, D. Van Gucht, G. H. L. Fletcher, Structural characterizations of the semantics of XPath as navigation tool on a document, in: Proceedings of the Twenty-Fifth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS 2006), Chicago, IL, USA, pp. 318–327.

[2] A. Tarski, On the calculus of relations, J. Symb. Log. 6 (1941) 73–89.

[3] R. Hirsch, I. Hodkinson, Relation Algebras by Games, volume 147 of *Studies in Logic and the Foundations of Mathematics*, Elsevier, Amsterdam, 2002.

[4] S. Givant, The calculus of relations as a foundation for mathematics, J. Autom. Reasoning 37 (2006) 277–322.

[5] R. D. Maddux, Relation Algebras, volume 150 of *Studies in Logic and the Foundations of Mathematics*, Elsevier, Amsterdam, 2006.

[6] A. Tarski, S. Givant, A formalization of set theory without variables, volume 41 of *Colloquium Publications*, American Mathematical Society, Providence, Rhode Island, 1987.

[7] M. Gyssens, L. V. Saxton, D. Van Gucht, Tagging as an alternative to object creation, in: J. C. Freytag, D. Maier, G. Vossen (Eds.), Query Processing for Advanced Database Systems, Morgan Kaufmann, San Mateo, CA, USA, 1994, pp. 201–242.

[8] B. ten Cate, M. Marx, Navigational XPath: calculus and algebra, SIGMOD Rec. 36 (2007) 19–26.

[9] M. Marx, M. de Rijke, Semantic characterizations of navigational XPath, SIGMOD Rec. 34 (2005) 41–46.

[10] G. H. L. Fletcher, M. Gyssens, D. Leinders, J. Van den Bussche, D. Van Gucht, S. Vansummeren, Y. Wu, The impact of transitive closure on the boolean expressiveness of navigational query languages on graphs, in: Proceedings of the 7th International Symposium on Foundations of Information and Knowledge Systems (FoIKS 2012), Kiel, Germany, pp. 124–143.

[11] G. H. L. Fletcher, M. Gyssens, D. Leinders, J. Van den Bussche, D. Van Gucht, S. Vansummeren, Y. Wu, Relative expressive power of navigational querying on graphs, in: Proceedings of the 14th International Conference on Database Theory (ICDT 2011), Uppsala, Sweden, pp. 197–207.

[12] G. H. L. Fletcher, D. Van Gucht, Y. Wu, M. Gyssens, S. Brenes, J. Paredaens, A methodology for coupling fragments of XPath with structural indexes for XML documents, Inf. Syst. 34 (2009) 657–670.

[13] V. M. Sarathy, L. V. Saxton, D. Van Gucht, Algebraic foundation and optimization for object based query languages, in: Proceedings of the Ninth IEEE International Conference on Data Engineering (ICDE 1993), Vienna, Austria, pp. 81–90.

[14] J. Hidders, J. Paredaens, XPath/XQuery, in: L. Liu, M. T. Özsu (Eds.), Encyclopedia of Database Systems, Springer US, 2009, pp. 3659–3665.

[15] M. Benedikt, W. Fan, G. M. Kuper, Structural properties of XPath fragments, Theor. Comput. Sci. 336 (2005) 3–31.

[16] G. Gottlob, C. Koch, Monadic queries over tree-structured data, in: Proceedings of the 17th Annual IEEE Symposium on Logic in Computer Science (LICS 2002), Copenhagen, Denmark, pp. 189–202.

[17] M. Benedikt, C. Koch, XPath leashed, ACM Comput. Surv. 41 (2009) 3:1–3:54.

[18] G. Gottlob, C. Koch, R. Pichler, Efficient algorithms for processing XPath queries, ACM Trans. Database Syst. 30 (2005) 444–491.

[19] M. Benedikt, W. Fan, F. Geerts, XPath satisfiability in the presence of DTDs, J. ACM 55 (2008).

[20] M. Bojanczyk, A. Muscholl, T. Schwentick, L. Segoufin, Two-variable logic on data trees and XML reasoning, J. ACM 56 (2009).

[21] G. Miklau, D. Suciu, Containment and equivalence for a fragment of XPath, J. ACM 51 (2004) 2–45.

[22] F. Bancilhon, On the completeness of query languages for relational data bases, in: Proceedings of the 7th Symposium on Mathematical Foundations of Computer Science (MFCS 1978), Zakopane, Poland, pp. 112–123.

[23] J. Paredaens, On the expressive power of the relational algebra, Inf. Process. Lett. 7 (1978) 107–111.

[24] A. K. Chandra, D. Harel, Computable queries for relational data bases, Journal of Computer and System Sciences 21 (1980) 156 – 178.

[25] D. Sangiorgi, J. Rutten, Advanced Topics in Bisimulation and Coinduction, Cambridge University Press, New York, NY, USA, 2011.

[26] B. ten Cate, T. Litak, M. Marx, Complete axiomatizations for XPath fragments, Journal of Applied Logic 8 (2010) 153 – 172.

[27] P. Buneman, M. Grohe, C. Koch, Path queries on compressed XML, in: Proceedings of the 29th International Conference on Very Large Data Bases (VLDB 2003), Berlin, Germany, pp. 141–152.

[28] I. Fundulaki, M. Marx, Specifying access control policies for XML documents with XPath, in: Proceedings of the ninth ACM Symposium on Access control models and technologies (SACMAT 2004), Yorktown Heights, New York, USA, pp. 61–69.

[29] R. Kaushik, P. Shenoy, P. Bohannon, E. Gudes, Exploiting local similarity for indexing paths in graph-structured data, in: Proceedings of the 18th IEEE International Conference on Data Engineering (ICDE 2004), San Jose, CA, USA, pp. 129–140.

[30] T. Milo, D. Suciu, Index structures for path expressions, in: Proceedings of the 7th International Conference on Database Theory (ICDT 1999), Jerusalem, Israel, pp. 277–295.

[31] Y. Wu, D. Van Gucht, M. Gyssens, J. Paredaens, A study of a positive fragment of path queries: Expressiveness, normal form and minimization, Comput. J. 54 (2011) 1091–1118.

[32] L. Libkin, Elements of finite model theory, Springer, Berlin, 2004.

[33] M. Marx, Conditional XPath, ACM Trans. Database Syst. 30 (2005) 929–959.

[34] L. Krzeszczakowski, Pebble games on trees, in: Proceedings of the 17th International Workshop on Computer Science Logic (CSL 2003), Vienna, Austria, pp. 359–371.

[35] M. Grohe, Equivalence in finite-variable logics is complete for polynomial time, Combinatorica 19 (1999) 507–532.

[36] S. Brenes Barahona, Structural summaries for efficient XML query processing, Ph.D. thesis, Indiana University, Bloomington, USA, 2011.